Vision-based 3D Pose Retrieval and Reconstruction

Dissertation

zur

Erlangung des **Doktorgrades (Dr. rer. nat.)**

der

Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

M.Sc. (Computer Science)
Hashim Yasin
aus Faisalabad, Pakistan

December 2015

Universität Bonn
Institut für Informatik II
Bonn, Germany

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn.

Erstgutachter: Prof. Dr. Andreas Weber

Zweitgutachter: Prof. Dr. Juergen Gall

Tag der Promotion: 17.03.2016

Erscheinungsjahr: 2016

Abstract

The people analysis and the understandings of their motions are the key components in many applications like sports sciences, biomechanics, medical rehabilitation, animated movie productions and the game industry. In this context, retrieval and reconstruction of the articulated 3D human poses are considered as the significant sub-elements. In this dissertation, we address the problem of retrieval and reconstruction of the 3D poses from a monocular video or even from a single RGB image. We propose a few data-driven pipelines to retrieve and reconstruct the 3D poses by exploiting the motion capture data as a prior. The main focus of our proposed approaches is to bridge the gap between the separate media of the 3D marker-based recording and the capturing of motions or photographs using a simple RGB camera. In principal, we leverage both media together efficiently for 3D pose estimation. We have shown that our proposed methodologies need not any synchronized 3D-2D pose-image pairs to retrieve and reconstruct the final 3D poses, and are flexible enough to capture motion in any studio-like indoor environment or outdoor natural environment.

In first part of the dissertation, we propose model based approaches for full body human motion reconstruction from the video input by employing just 2D joint positions of the four end effectors and the head. We resolve the 3D-2D pose-image cross model correspondence by developing an intermediate container the *knowledge base* through the motion capture data which contains information about how people move. It includes the 3D normalized pose space and the corresponding synchronized 2D normalized pose space created by utilizing a number of virtual cameras. We first detect and track the features of these five joints from the input motion sequences using SURF, MSER and colorMSER feature detectors, which vote for the possible 2D locations for these joints in the video. The extraction of suitable feature sets from both, the input control signals and the motion capture data, enables us to retrieve the closest instances from the motion capture dataset through employing the fast searching and retrieval techniques. We develop a graphical structure *online lazy neighbourhood graph* in order to make the similarity search more accurate and robust by deploying the temporal coherence of the input control signals. The retrieved prior poses are exploited further in order to stabilize the feature detection and tracking process. Finally, the 3D motion sequences are reconstructed by a non-linear optimizer that takes into account multiple energy terms. We evaluate our approaches with a series of experiment scenarios designed in terms of performing actors, camera viewpoints and the noisy inputs. Only a little preprocessing is needed by our methods and the reconstruction processes run close to real time.

The second part of the dissertation is dedicated to 3D human pose estimation from a monocular single image. First, we propose an efficient 3D pose retrieval strategy which leads towards a novel data driven approach to reconstruct a 3D human pose from a monocular still image. We design and devise multiple feature sets for global similarity search. At runtime, we search for the similar poses from a motion capture dataset in a definite feature space made up of specific joints. We introduce two-fold method for camera estimation, where we exploit the view directions at which we perform sampling of the MoCap dataset as well as the MoCap priors to minimize the projection error. We also benefit from the MoCap priors and the joints' weights in order to learn a low-dimensional local 3D pose model which is constrained further by multiple energies to infer the final 3D human pose. We thoroughly evaluate our approach on synthetically generated examples, the real internet images and the hand-drawn sketches. We achieve state-of-the-arts results when the test and MoCap data are from the same dataset and obtain competitive results when the motion capture data is taken from a different dataset.

Second, we propose a dual source approach for 3D pose estimation from a single RGB image. One major challenge for 3D pose estimation from a single RGB image is the acquisition of sufficient training data. In particular, collecting large amounts of training data that contain unconstrained images and are annotated with accurate 3D poses is infeasible. We therefore propose to use two independent training sources. The first source consists of images with annotated 2D poses and the second source consists of accurate 3D motion capture data. To integrate both sources, we propose a dual-source approach that combines 2D pose estimation with efficient and robust 3D pose retrieval. In our experiments, we show that our approach achieves state-of-the-art results and is even competitive when the skeleton structure of the two sources differ substantially.

In the last part of the dissertation, we focus on how the different techniques, developed for the human motion capturing, retrieval and reconstruction can be adapted to handle the quadruped motion capture data and which new applications may appear. We discuss some particularities which must be considered during capturing the large animal motions. For retrieval, we derive the suitable feature sets in order to perform fast searches into the MoCap dataset for similar motion segments. At the end, we present a data-driven approach to reconstruct the quadruped motions from the video input data.

Acknowledgements

Foremost, financial support extended by the Higher Education Commission (HEC) of Pakistan in completing the research, logistic support, *etc.* is gratefully acknowledged.

I would like to express my great appreciation to my supervisor Prof. Dr. Andreas Weber for his complete support in this research work and preparing this dissertation. He always guided me in unseen and unexplored areas of research and made me feel easy to grope through the difficult ways of the field of research. Many heartfelt thanks to Prof. Dr. Andreas Weber.

I would also like to offer my special thanks to my second supervisor Prof. Dr. Juergen Gall who provided a thought-provoking source and a complete academic and technical guidance in carrying out this research project. His overall suggestions and time to time hints saved a lot of time and smoothened the way to a successful completion of the project. I am also very thankful to the other committee members Prof. Dr. Joachim Anlauf and Prof. Dr. Klaus Desch who agree to join as the committee member.

Last but not least, I don't have words to express my feelings and thanks to my late parents who always encouraged and inspired me for higher studies. Their selfless affection, cordial attitude, and effective motivations always remain for me a torch of light and guidance in every walk of life. I also acknowledge my other family members who supported me in sparing me beyond office hours in smooth execution of this project. At the end, I am particularly grateful for the support and good times given by my colleagues and friends.

Contents

List of Figures

List of Tables

Abbreviations

MoCap	Motion Capture dataset
TOF	Time-of-flight sensor
SURF	Speeded Up Robust Features
MSER	Maximally Stable Extremal Regions
PCA	Principal Component Analysis
IMUs	Inertial Measurement Units
K**nn**	K nearest neighbor
K_w**nn**	Weighted K nearest neighbor
SO	Special Orthogonal group
SE	Special Euclidean group
SS	Saliency Scale
SIFT	Scale-Invariant Feature Transform
DoG	Difference-of-Gaussian
LoG	Laplacian-of-Gaussian
HOG	Histograms of Oriented Gradients
SVM	Support Vector Machine
HSV	Hue,Saturation, Value
PSM	Pictorial Structure Model
DCT	Discrete Cosine Transform
PLCR	Pose Locality Constrained Representation
OCSVM	one-class Support Vector Machine
OLNG	Online Lazy Neighbourhood Graph
DTW	Dynamic Time Warping
RBF	Radial Basis Function
DoF	Degree of Freedom
DOFs	Dictionary of Features
CNN	Convolution Neural Network
OMP	Orthogonal Matching Pursuit
ANN	Approximate Nearest Neighbor
LNG	Lazy Neighbourhood Graph

Symbols

Symbol	**Meaning**
$\mathbf{X}$	3D positions of the pose
$\mathbf{x}$	2D positions of the pose
$\widetilde{\mathbf{x}}$	2D projection of the 3D pose
$\hat{\mathbf{x}}$	Refined 2D pose
X_i	The i^{th} joint position of 3D pose, $X_i = [\mathsf{X}_{x,i}, \mathsf{X}_{y,i}, \mathsf{X}_{z,i}, 1]^T \in \mathbb{R}^4$, in homogenous coordinates
x_i	The i^{th} joint position of 2D image-based pose, $x_i = [\mathsf{x}_{x,i}, \mathsf{x}_{y,i}, 1]^T \in \mathbb{R}^3$ in homogenous coordinates
$\mathcal{F}_5^{\text{3D}}$	3D feature sets
$\mathcal{F}_5^{\text{syn}}$	2D feature sets extracted from the MoCap dataset
$\mathcal{F}_5^{\text{vid}}$	2D feature sets extracted from a video
$\mathcal{F}_5^{\text{im}}$	2D feature sets extracted from an image
$\mathcal{F}_{\mathcal{J}}$	Feature sets with specified joints
$\mathcal{J}$	The joints included in skeleton model
$\mathcal{J}_{\mathcal{F}}$	The joints involved in feature set
$\mathcal{J}_s$	The joint set with $s \in \{up, lw, lt, rt, all\}$
$\hat{s}$	Inferred joint set
$\mathcal{G}$	A graph that represents PSM with vertices $\mathcal{J}$ corresponding to joints and edges $\mathcal{L}$ corresponding to segments
$\mathbf{\Psi}$	3D normalized pose space
ψ	2D normalized pose space
$\mathbf{R}_{(\alpha,\beta,\gamma)}$	Rotation with 3 rotational parameters (α, β and γ)
$\mathbf{T}_{(x,y,z)}$	Translation with translational parameters (T_x, T_y and T_z)
$\mathcal{M}$	Projection matrix
$\hat{\mathcal{M}}$	Inferred projection matrix
$\mathbf{W}$	Intrinsic camera parameters
$\mathcal{E}$	Error representation
$\mathbb{Q}$	Quaternion pose space
$\mathbf{Q}$	Joint angle configurations of the pose
$\widetilde{\mathbf{Q}}$	Joint angle configurations of the synthesized pose
$\widetilde{\mathbf{X}}$	Positional data of the synthesized pose
$\widehat{\mathbf{Q}}$	Joints angle configuration of the reconstructed pose
$\widehat{\mathbf{X}}$	Positional data of the reconstructed pose
$\mathbf{S}$	Skeleton model
$\mathbf{f}$	Forward kinematics function

K	Number of nearest neighbors
K_w	Weighted nearest neighbors
$\mathcal{H}$	Indices for nearest neighbors
N	Total number of frames of the MoCap dataset
M	Total number of frames of the query
P	Total number of 2D projections (number of virtual cameras)
$\mathcal{Z}$	The size of the window
$\mathbf{I}$	Image
E	The energy representation
$\mathcal{U}$	Camera view directions
$\mathcal{V}$	Camera view directions observed from sampling of MoCap data
$\mathbf{w}$	Joints' weights
$\mathcal{DB}$	Database
$\mathcal{TS}$	Test dataset
P	Probability
n	Axis of rotation
$\widehat{n}$	Screw symmetric matrix of n
ξ	Twist representation

This dissertation is a very humble dedication

to my beloved

Holy Prophet Muhammad صلى الله عليه وسلم *(peace be upon Him),*

Who is the fountain of knowledge, the forever source of learning and the Divine light to illuminate the darkness of ignorance.

1

Introduction

The true sign of intelligence is not knowledge but imagination. Logic will get you from A to B. Imagination will take you everywhere.

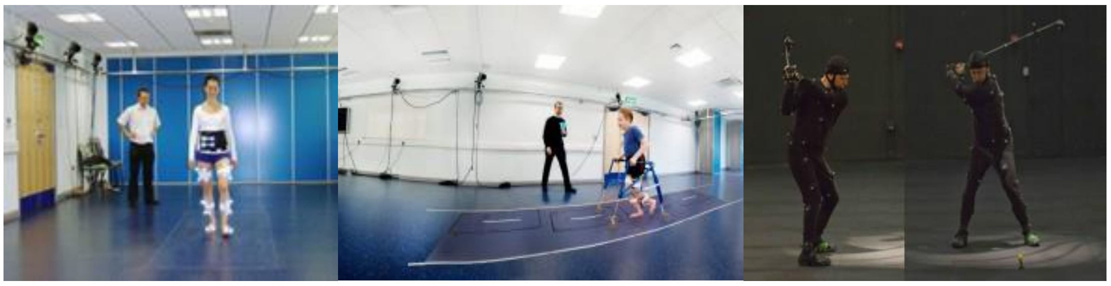
Albert Einstein

The people analysis and the synthesizing of qualitative 3D human poses from the videos or images, have been studied and investigated from the last few decades, but still an intensive interest has been seemed in this area of human motion understanding, analysis and 3D reconstruction. As a result, the demand for high quality motion capture data is increasing and new applications for everywhere motion capture, based on various consumer electronic devices, are emerging. In this dissertation, we present a few pipelines, algorithms and techniques to synthesize the plausible 3D pose from a sparse control input signal that is whether in the form of a video or even a single monocular image.

1.1 Motivation

Motion is always considered as an important cue for analysing the people in terms of understanding their activities and gestures. The people analysis, the 3D synthesizing of human poses and the gesture recognition exploiting some motion capture (MoCap) data have many potential applications such as,

- *Biomechanics and medical rehabilitation.* Motion analysis via the MoCap data is an essential component in producing valuable biomechanical data which is deployed further for clinical gait analysis, injury detection and many orthopedic applications *i.e.*: prosthetic designs; the simulation and modeling of mechanical properties of the bones and soft tissues; spine analysis and the body joint mechanics *etc.* [Men10]. Figure 1.1(a)–(b) show a few examples of clinical biomechanics used for the purpose of clinical gait analysis. The acquisition of biomechanical data and its use in medical rehabilitation is equally important and essential for the treatment of domestic animals as well.



(a) Clinical biomechanics **(b)** Clinical gait analysis **(c)** Sports analysis

Figure 1.1: Human motion capture for medical rehabilitation and sports analysis[1].

- *Entertainment.* Animation of the human like characters as well as the non-humanoid characters by employing the motion capture library is extensively used in animated movies industry. A few examples of the human like animated virtual characters of some popular movies like *Avatar*, *The Lord of the Rings* and *The Polar Express* are presented in Figure 1.2(a)-(c) respectively. Such types of productions are completely dependent on high quality motion capture data through which the human like or the non-humanoid virtual characters are generated using computer graphics techniques.

- *Surveillance systems.* Automated surveillance systems are of great practical importance especially in security sensitive areas *e.g.* foreign offices and the airports. These systems monitor human activities in realtime and necessarily require some techniques and methods to be capable of detecting anomalous and suspicious human movements and actions.

- *Robotics.* In robotics, the anticipating and predicting 3-dimensional dynamic human activities as well as the close observations of these actions in 3D are the crucial elements for realtime interactive manipulation to the environment and the obstacle avoidance.

- *Sports analysis.* Another important application of the motion capture data is the sports analysis, where the 3D motion capture data is used in order to enhance the player's overall performance through modeling and simulation of motions. Such techniques are quite popular especially in tennis, baseball, golf and gymnastic *etc.* [Men10]. The biomechanics of the swing during playing golf is shown in Figure 1.1 (c) that is utilized to improve the player's efficiency.

[1]Image Sources: (a)-(b) http://www.qualisys.com (c) http://www.moenormangolf.com (visited on September 2015)

(a) Avatar

(b) The Lord of the Rings

(c) The Polar Express

Figure 1.2: Marker-based motion capture for human-like animated characters used in different animated movies[2].

- *Game industry.* The motion capture data, either the human motions based or the quadruped motions based, is also very popular in generating the human and the non-humanoid motions in games *e.g. L.A. Noir.*

1.2 Why Vision-based 3D Poses?

On one hand, for generating the 3D human poses, the virtual marker based motion capture system is very popular and has become a standard technique to record human motions, like *Vicon*, *MX* and *Giant etc.* There are a growing pool of high-quality motion capture datasets, which are publically available and are used for research and scientific studies [MRC^{+}07, CMU14, GFB12]. On the other hand, it requires highly expensive indoor hardware set up. This studio like environment and the attached markers prevent the capturing of realistic videos and images. Alternatively, RGB depth Kinect cameras are utilized which seems to be more convenient comparatively, but unfortunately Kinect cameras have limited operational range and are capable for recording depth information within the range of 1.0m to 4.0m. Moreover, Kinect RGB cameras are not feasible in outdoor environment due to the reason that Kinect infrared radiations get interrupted by sunlight.

Despite of having aforementioned motion capture technologies, we still have to deal with a bundle of internet, television and movie's videos/images, which contain no any depth information at all. The surveillance systems mostly make use of RGB cameras which do not record depth information as well. Furthermore, in practice, the 3D human

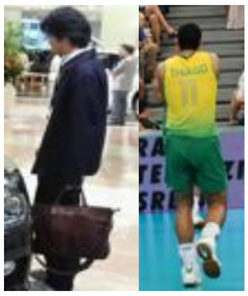
[2]Image Sources: (a) http://www.davidbordwell.net (b) http://www.serkis.com (c) http://www.calvin.edu (visited on September 2015)

motion capture is hard enough, even more than that it seems to be in case of the virtual marker based standard motion capture systems or RGB depth Kinect cameras. These systems require a significant postprocessing in order to handle the missing information for generating the accurate captured posture [PMFR14]. Hence, as a solution, a bulk of research has been conducted to estimate the 3D poses from videos/images, but there are still many open challenges that need to be tackled in order to meet the massive demands of the motion capture data. Additionally, the vision-based motion capture and accurate prediction of the 3D poses from video/images empower many vision-dependent applications such as surveillance systems, robotics, entertainment and health rehabilitation industries.

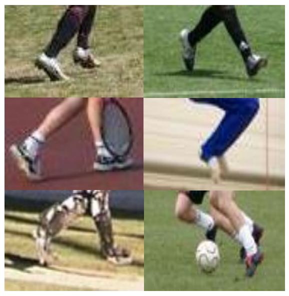
1.3 Challenges

The articulated 3D human pose estimation from a monocular single video or an image is an under-constrained and ill-posed problem. The challenges in 3D human pose estimation can be summarized as follows;

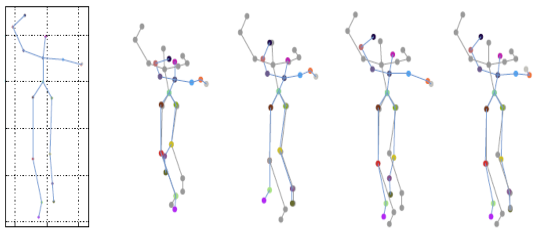
- There may exist several plausible 3D articulated poses for a single 2D image pose. Multiple 3D poses with different orientations may project on the same single 2D image pose (see Figure 1.3 (c)). As a result, to infer accurate 3D pose just on the basis of 2D joint locations become a challenging task.
- During image capturing process, the depth information is lost that may be due to the lens distortion, camera movements, noise or image resolution. These factors limits the chances to estimate accurate 3D pose.
- The existence of irreversible perspective projection or the missing camera parameters become hindrance in 2D-3D correspondence and make it harder to recover the plausible 3D pose.
- The occlusion—where a few body parts of a person are occluded by some objects or by its own body, called *self-occlusion* (see Figure 1.3(a)); image blurring; viewpoints variations; different clothing, shapes and backgrounds; and the illumination effects may create hindrance in identifying correct 2D joint locations. These artifacts not only yield ambiguities in 2D pose inference in an image but also lead towards obscure and uncertain 3D pose estimation.
- In 2D pose estimation, the symmetric parts (like knees, feet, elbows and shoulders *etc.*) of a human body can cause the flipping of left and right sides as well as the double counting error—where the pairs of body parts are detected on the same



(a)



(b)



(c)

Figure 1.3: Example images that demonstrate a few challenges for pose estimation; **(a)** Occlusion, **(b)** Left/right side ambiguities and **(c)** Multiple 3D pose interpretations for a single 2D pose, gray dots are the ground truth and color dots are 3D poses at different views [AB15].

location. A few images for the ambiguous left and right sides of the lower body parts are shown in Figure 1.3(b). Consequently, the 3D pose estimation become more tough and difficult.

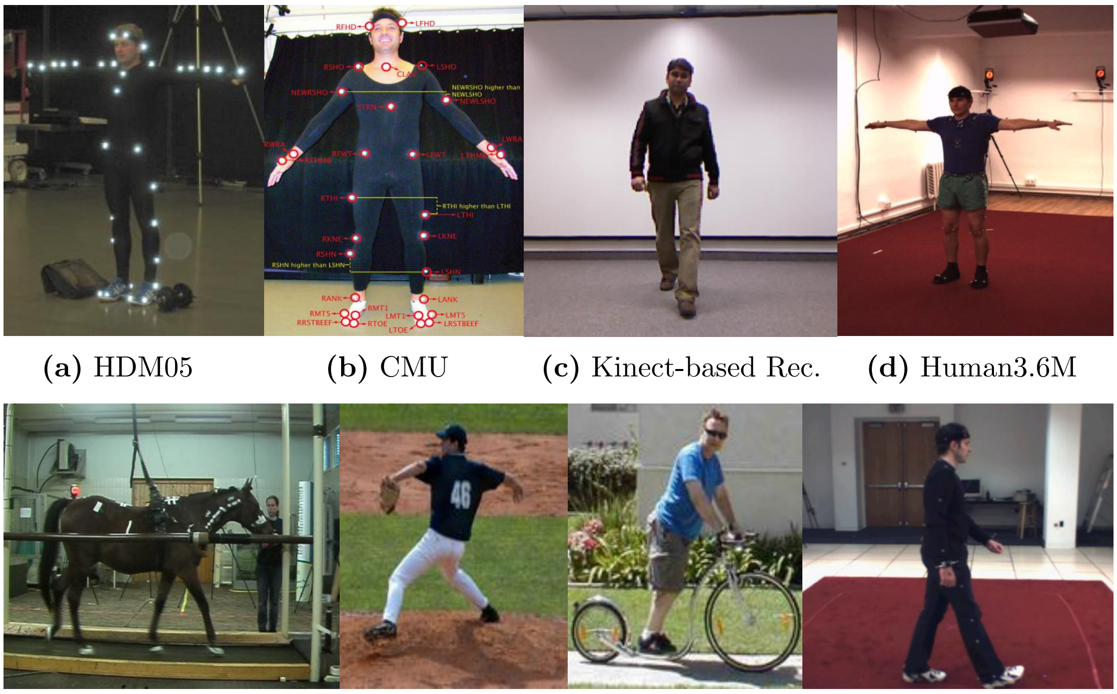
1.4 Evaluation datasets

There are a very few publically available datasets which have both, the 3D motion capture data and the synchronized videos or images with full camera parameters. The evaluation datasets (MoCap datasets as well as the testing datasets consisting of input videos/images) which we employ in this dissertation to conduct a series of different experiments are described one by one as below. Some example images for these datasets are shown in Figure 3.4.

1.4.1 CMU Motion Capture Dataset

CMU motion capture dataset is the huge dataset containing diverse collection of human motion categories but unfortunately it does not contain the relevant synchronized recorded videos for each motion type. There are videos of a few motion types but these videos neither have corresponding synchronized 3D motion capture data nor any detail about camera parameters. Motion is captured using 12 Vicon infrared MX-40 cameras which record motions with sampling rate 120 Hz. The performing actors wear a black suit with 41 markers taped on [CMU14]. We benefit from this dataset in two ways,

- We utilize this MoCap dataset as pre-existing prior knowledge in our proposed frameworks in order to resolve the missing depth information.
- We generate synthetic 2D image-based testing datasets using MoCap files.



(a) HDM05 (b) CMU (c) Kinect-based Rec. (d) Human3.6M

(e) Horse MoCap (f) Leeds Sports (g) PARSE (h) HumanEva-I

Figure 1.4: Example images for different datasets used in this dissertation for the purpose of performance evaluations of the proposed approaches.

1.4.2 HDM05 Motion Capture Dataset

In recording of HDM05 motion capture dataset, 40-50 retro-reflective markers are attached with the performing actor's suit and these markers are trapped and captured by using Vicon MX system with 12 high-resolution cameras at a sampling rate of 120 Hz. It consists of roughly 70 motion classes executed by various performing actors and as a result, it contains roughly 1500 motion clips [MRC$^+$07]. This dataset is also deficient in synchronized videos corresponding to the 3D motion capture data as well as the camera parameters. We utilize this dataset in the same way as in case of CMU motion capture dataset.

1.4.3 Human3.6M Dataset

Human 3.6M dataset consists of the 3D motion capture data (3.6 million 3D human poses) synchronized with corresponding videos as well using hardware and software synchronization with corresponding sensors. For capturing motion data, they use 15 sensors *i.e.* 4 digital video cameras with frame rate 50 Hz and resolution 1000 × 1000, 1 time-of-flight (TOF) sensor with 25 Hz, 10 Vicon T40 cameras with frame rate 200

Hz [IPOS14]. They also provide accurate background subtraction, full camera parameters and the persons' bounding boxes. Total 11 professional actors (6 male, 5 female) perform 15 different activities. We exploit this dataset for quantitative evaluations.

1.4.4 HumanEva-I Dataset

In HumanEva-I dataset, the human motion is captured through reflective markers and 6 1M-pixel cameras. For video data acquisition, 3 color cameras and 4 grayscale cameras are utilized [SBB10]. Afterward, the 3D articulated poses and the corresponding videos are synchronized by software synchronization process. They employ 4 subjects that perform 6 different motion categories. We deploy this dataset for quantitative evaluations in Chapter 7.

1.4.5 Leeds Sports Pose Dataset

This dataset consists of 2000 pose-annotated images taken from outdoor environment in real scenarios. It splits into two parts: 1000 images for training and 1000 images for testing. Images are annotated with utilizing 14 joints [JE10]. We employ this dataset for qualitative evaluations of our proposed image-based 3D pose estimation framework.

1.4.6 PARSE Dataset

This dataset contains 305 images with people of roughly 150 pixels in height. The first 100 images of the dataset are used for training, while the remaining 205 images are used for testing purpose [Ram07]. We use this dataset for qualitative evaluation only.

1.4.7 Kinect-based Recording

We also record video sequences of different activities like: grabbing, walking, jumping jack, jogging *etc.* that are performed by four different actors, using Kinect RGB cameras in indoor environment. We use these videos in our video-based reconstruction approach for qualitative analysis. An example image of Kinect-based video recording is shown in Figure 3.4 (c).

1.4.8 Horse MoCap Dataset

The horse MoCap dataset consists of the motion sequences of five horses where each horse performs two motion activities *e.g.* walk and trot on a treadmill. Each motion action has been performed with at least three trials (each of which is about 10 seconds). As a results, the horse MoCap dataset contains 30 motion trials with varying numbers of motion cycles. There are total 36,000 number of frames with a sampling rate of 120 Hz which corresponds to five minutes of motion capture data. The horse dataset also has the corresponding video sequences which are not synchronized with the 3D horse MoCap data. The more details about dataset will be discussed in Chapter 8.

1.5 Contributions

This dissertation deals with the vision-based 3D pose retrieval and reconstruction when the input to the system is whether in the form of a monocular real/synthetic video or a single real RGB/synthtic image. The contribution of the dissertation can be explained as follows;

- **Video based 3D Reconstruction of Human Motions.** The first part of the dissertation is dedicated to 3D human motion reconstruction from video input data, where we propose novel data-driven frameworks for 3D motion retrieval and reconstruction from video sequences. In first proposed model based framework, we retrieve and reconstruct 3D full body human motion through the MoCap data on the basis of 2D locations of just five joints (the four end effectors and the head). We locate and track these five joints by local feature detectors/descriptors like SURF (Speeded Up Robust Features) integrated with MSER (Maximally Stable Extremal Regions) through developing a dictionary of features (DOFs). For efficient retrieval of nearest neighbors from the MoCap dataset, we introduce *knowledge base* which consists of 3D normalized pose space and the corresponding synchronized 2D normalized pose space developed through sampling of MoCap data at several virtual cameras. With the help of pre-existing prior knowledge available in the MoCap dataset, we construct a 3D pose model in low dimensional Principal Component Analysis (PCA) subspace. For final 3D pose inference, we make the model to fit in accordance with the MoCap priors as well as map it onto image features through projective constraint. We evaluate the proposed system on synthetic and the real videos as well as testify the influence of performing actor, virtual cameras, testing camera viewpoints and the impact of noisy input on overall system's performance.

  In second methodology, we enhance the similarity search and retrieval method by utilizing the temporal coherence of the input control signal by developing a directed acyclic graphical structure *online lazy neighbourhood graph* (OLNG) and as a result we obtain the optimal nearest neighbors and weight them by considering costs associated with the paths in OLNG. We exploit these optimal and weighted 3D poses in final 3D motion reconstruction, where we formulate non-linear objective function with multiple energy terms exploiting symmetric square root kernel to measure the probability density, which is optimized by gradient decent based algorithm. We further benefit from these optimal 3D examples to make detection and tracking more robust and efficient.

- **Recovering 3D Pose from an Image.** In the second part of the dissertation, we have considered the most challenging scenario like 3D pose estimation from a single image either synthetic or real. We design multiple feature sets for efficient pose similarity search into the MoCap dataset and propose a pipeline for data-driven 3D pose estimation from a subset of 2D joint positions. At runtime, a nearest neighbor search is performed to retrieve similar poses through developed *knowledge base*. These closest examples are used to estimate camera parameters and are exploited in proposed reconstruction approach for final 3D human pose prediction. We also derive benefits from joints' degree of freedom to make reconstruction efficient. We thoroughly evaluate our approach on 2D synthetic examples, the real images and the hand-drawn sketches. Our proposed approach executes state-of-the-art results when the test data is generated from same MoCap dataset and even produces competitive results when a different MoCap dataset is employed.

- **A Dual Source Approach for 3D Pose Estimation from a Single Image.** We propose a dual source approach in order to estimate 3D human pose from a single image, where we leverage image-based training source and MoCap data together to get benefits from both media. During inference, we estimate the 2D pose from a given RGB image and retrieve the nearest 3D poses from MoCap dataset using an approach that is robust to 2D pose estimation errors. We then estimate a mapping from the 3D pose space to the image and weight the retrieved poses according to the image evidence. From the weighted poses, a 3D pose model is constructed and fit to the image in order to estimate the 3D pose. During this process, the 2D pose is also refined through exploiting the retrieved 3D poses and as a result the approach can be iterated for further refinement in 3D pose estimation. For evaluation, we employ two popular datasets and on both datasets, our proposed approach achieves state-of-the-art results. We also analyze the influence of the skeleton discrepancies between the two training sources as well as the impact of the accurate 2D inputs on the performance of the proposed approach.

- **Quadruped Motions: Retrieval and Reconstruction.** In the last part of the dissertation, we deal with retrieval and reconstruction of quadruped motions and demonstrate that how the human motion capture, retrieval and reconstruction techniques can be adapted efficiently to quadruped motions. We illustrate that how to handle quadruped motion capture data and elaborate some particularities for suitable markers' setup. We design several 3D and 2D feature sets on which basis we retrieve the similar poses from the horse MoCap data and predict 3D quadruped motion directly in Euclidean space through these retrieved nearest examples by developing an energy function.

1.6 Thesis Outlines

- **Chapter 1. Introduction.** Introduction chapter contains details about motivation, problem statement, challenges, evaluation datasets and the dissertation's contributions.
- **Chapter 2. Body Models.** This chapter focuses on the pose representations as well as the body structure models. We discuss in details about the different pose parameterizations and the body models for both 2D/3D data.
- **Chapter 3. Related Work.** In this chapter, we will provide an overview of the state-of-the-art approaches relevant to 3D human pose estimation.

Part-I 3D Motion Reconstruction from Video

- **Chapter 4. 3D Motion Reconstruction from Video.** This chapter introduces the proposed model based framework for 3D retrieval and reconstruction of full body human motions from video input data. We will illustrate that how can we retrieve nearest neighbors from the MoCap dataset utilizing a subset of 2D joints efficiently and how these nearest neighbors can be exploited in ultimate 3D motion reconstruction.
- **Chapter 5. 3D Motion Tracking and Reconstruction.** In this chapter, the video-based motion reconstruction framework is presented where we will describe the details about how the temporal information can be utilized in retrieval and reconstruction methodologies. We will also elaborate that how the video-based feature detection and tracking can be made robust through the MoCap priors. Moreover, we will discuss the kernel-based approach to estimate probability density of the MoCap priors for final 3D motion reconstruction.

Part-II 3D Pose Estimation from a Monocular Single Image

- **Chapter 6. Recovering 3D Pose from an Image.** The chapter describes a pipeline to estimate 3D human pose from a single 2D image pose either synthetic or real. We will illustrate different feature sets that we use for efficient retrieval of similar poses from the MoCap dataset and we will also explore that how the retrieved poses can be used to estimate camera parameters and to recover final 3D pose.

- **Chapter 7. A Dual-Source Approach for 3D Pose Estimation from a Single Image.** In this chapter, we will provide details about the proposed dual source approach to infer 3D human pose from monocular single image. Here, we will explain that how can we get benefits from the 2D annotated training sources as well as the 3D motion capture library to predict the final 3D human pose from a single image.

Part-III Quadruped Motions: Retrieval and Reconstructions

- **Chapter 8. Retrieval and Reconstruction of Quadruped Motions.** This chapter deals with the retrieval and reconstruction of quadruped (horse) motions. We will discuss different retrieval strategies and the feature sets with their efficiencies.

- **Chapter 9. Conclusion and Future Perspectives.** At the end, we will summarize all the proposed approaches presented in this dissertation and will provide the conclusive remarks. Future directions and perspectives for these proposed approaches will also be discussed in this chapter.

2

Body Models

Since we cannot know all that there is to be known about anything, we ought to know a little about everything.

Blaise Pascal

In this chapter, we discuss the preliminary concepts, the algorithms and the techniques that we consider for the 3D/2D pose representations and for developing the body structure model in order to accomplish the tasks of the 3D pose retrieval and reconstruction. First, we elaborate the details about the pose representation in Section 2.1. The remaining chapter is structured as: in Section 2.2, we discuss briefly the different body models found in the literature and the body structures which we have employed, we explores the 2D pose representations in Section 2.3 and Section 2.4 provides details about the quadruped body models.

2.1 Pose Representations

There are a number of ways to represent the human pose *e.g.* through the joint angle configurations like the Euler angles, quaternions and axis-angle representations, or through the joint coordinates directly in Euclidean space. The full body human pose is represented by different rigid body segments that are linked together by different types of body joints. Mostly, the kinematic model is found in the literature to model the human body in a tree structure. The kinematic model consists of the pose parameters that are defined by the root joint's positions as well as the orientations with 6 degrees of freedom in the world coordinate system. A common parametrization for a human pose is the kinematic chain where the movement of a segment depends on the movement of the other body segments. For instance, the foot position is dependent on the position of the lower leg that is further dependent on the position of the upper leg. The movement of the segment with respect to its parent segment can be parameterized by rigid body transformation *e.g.* rotations and translations.

2.1.1 Joint Angle Configurations

Let $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix and is belong to the *special orthogonal* group $SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times3} | \mathbf{R}\mathbf{R}^T = \mathbf{R}^T\mathbf{R} = I, \det(\mathbf{R}) = 1\}$. Let $\mathbf{T} \in \mathbb{R}^3$ is the translation. The product of both determines the rigid body transformation and it is represented by the *special Euclidean* group $SE(3) = \mathbb{R}^3 \times SO(3)$.

Euler Angles. The standard method to represent the orientation of an object with respect to another object is in the form of Euler angles. A rotation matrix around the x, y and z axes can be represented as,

$$\mathbf{R}(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix}, \tag{2.1}$$

$$\mathbf{R}(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix}, \tag{2.2}$$

$$\mathbf{R}(\gamma) = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

where the angles α, β and γ are called the Euler angles. As matrix multiplication does not commute, therefor multiplication order may affect the final results. The rotation along the x-axis, then y-axis, and finally along z-axis can be represented as,

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}(\gamma) \cdot \mathbf{R}(\beta) \cdot \mathbf{R}(\alpha) \tag{2.4}$$

$$= \begin{bmatrix} \cos(\beta)\cos(\gamma) & \sin(\alpha)\sin(\beta)\cos(\gamma) - \cos(\alpha)\sin(\gamma) & \cos(\alpha)\sin(\beta)\cos(\gamma) + \sin(\alpha)\sin(\beta) \\ \cos(\beta)\sin(\gamma) & \cos(\alpha)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma) & \cos(\alpha)\sin(\beta)\sin(\gamma) - \sin(\alpha)\cos(\gamma) \\ -\sin(\beta) & \sin(\alpha)\cos(\beta) & \cos(\alpha)\cos(\beta) \end{bmatrix} \tag{2.5}$$

As a result, we can compute any rotation with these Euler angles (α, β, γ). Unfortunately, the Euler angles have an unsolved problem *gimbal lock*, in which one degree of freedom is lost. This happen when two rotation axes align to each other *e.g.* $\beta = 0$ or $\beta = 90$ *etc.*, then the solution is the quaternion.

Quaternions. The quaternion $\mathbf{Q}$ is spanned in 4 dimensional space $\mathbb{Q}$ by one real axis and three orthogonal axes and can represent rotation. The quaternion $\mathbf{Q} = (q_0, \mathbf{q})$ is a vector that consists of one scaler component q_0 and three vector components $\mathbf{q} =$

(q_1, q_2, q_3) and can be expressed as,

$$\begin{aligned} \mathbf{Q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}, &\quad \text{with,} \\ \mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{i}\cdot\mathbf{j}\cdot\mathbf{k} = 1, \\ \mathbf{i}\cdot\mathbf{j} = -\mathbf{j}\cdot\mathbf{i} = \mathbf{k}, \\ \mathbf{j}\cdot\mathbf{k} = -\mathbf{k}\cdot\mathbf{j} = \mathbf{i}, \\ \mathbf{k}\cdot\mathbf{i} = -\mathbf{i}\cdot\mathbf{k} = \mathbf{j}. \end{aligned} \tag{2.6}$$

The *conjugate* of the quaternion $\mathbf{Q} = (q_0, \mathbf{q})$ is defined as $\mathbf{Q}^* = (q_0, -\mathbf{q})$ and the magnitude and the inverse of the quaternion can be represented respectively as,

$$\|\mathbf{Q}\| = \sqrt{\mathbf{Q}\mathbf{Q}^*} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}, \tag{2.7}$$

$$\mathbf{Q}^{-1} = \frac{\mathbf{Q}^*}{\|\mathbf{Q}\|}. \tag{2.8}$$

The unit quaternions are the quaternions $\mathbf{Q} \in \mathbb{Q}$ with norm $\|\mathbf{Q}\| = 1$. The inverse of the unit quaternion is just the *conjugate* quaternion $\mathbf{Q}^{-1} = \mathbf{Q}^*$. The identity element for the quaternion multiplication is represented as $\mathbf{Q} = (1, 0)$ [MSZ94]. Given two quaternions $\mathbf{Q}_a = (q_{0,a}, \mathbf{q}_a)$ and $\mathbf{Q}_b = (q_{0,b}, \mathbf{q}_b)$, the multiplication between them can be computed as,

$$\mathbf{Q}_a\mathbf{Q}_b = q_{0,a}q_{0,b} - \mathbf{q}_a \cdot \mathbf{q}_b \ , \ q_{0,a}\mathbf{q}_b + q_{0,b}\mathbf{q}_a + \mathbf{q}_a \times \mathbf{q}_b. \tag{2.9}$$

The quaternion multiplication is distributive and associative, but not commutative. A rotation $\mathbf{R} = \exp(\widehat{n}\theta)$ about an axis $n \in \mathbb{R}^3$ with an angle $\theta \in \mathbb{R}$ can be represented by the quaternion as,

$$\mathbf{Q} = \left(\cos\left(\frac{\theta}{2}\right), n\sin\left(\frac{\theta}{2}\right)\right). \tag{2.10}$$

The rotation matrix from a quaternion with magnitude $\|\mathbf{Q}\| = 1$ can be computed [Sho85] as,

$$\mathbf{Q} = \begin{bmatrix} 1 - 2q_2^2 - 2q_3^2 & 2q_1q_2 + 2q_0q_3 & 2q_1q_3 - 2q_0q_2 \\ 2q_1q_2 - 2q_0q_3 & 1 - 2q_1^2 - 2q_3^2 & 2q_2q_3 + 2q_0q_1 \\ 2q_1q_3 + 2q_0q_2 & 2q_2q_3 - 2q_0q_1 & 1 - 2q_1^2 - 2q_2^2 \end{bmatrix}. \tag{2.11}$$

Axis-angle. We often deal with a situation when we need to describe the joint's movement along a specified axis and an angle. In this context, we need a unit vector $n \in \mathbb{R}^3$ that fix the direction of the rotation and the specified angle $\theta \in \mathbb{R}$ to represent the amount of rotation, then the net rotation can be expressed in exponential form as,

$$\mathbf{R} = \exp(\widehat{n}\theta), \tag{2.12}$$

where $(\wedge)$ is the wedge operator, $n^{\wedge} = \widehat{n} \in so(3)$ is the skew-symmetric matrix which verifies $so(3) = \{\mathbf{S} \in \mathbb{R}^{3\times3} | \mathbf{S}^T = -\mathbf{S}\}$, and has been computed from n as

$$n = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix}, \quad \widehat{n} = \begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix}. \tag{2.13}$$

Due to the screw symmetric matrix $\widehat{n}$, the exponential of the rotation matrix $\exp(\widehat{n}\theta)$ can be represented in a closed form as,

$$\exp(\widehat{n}\theta) = I + \widehat{n}\sin\theta + \widehat{n}^2(1 - \cos\theta). \tag{2.14}$$

This is known as *Rodrigues' formula* that is a simple form to compute the rotation matrix with axis of rotation n and the rotational angle θ.

For rigid body transformations, rotation and translation, the $n = (n_1, n_2, n_3)$ is extended to $\xi = (n_1, n_2, n_3, \upsilon_1, \upsilon_2, \upsilon_3)$ with difference in points $\upsilon = (\upsilon_1, \upsilon_2, \upsilon_3)$, and as a result, the twist is represented as,

$$\theta\widehat{\xi} = \theta \begin{bmatrix} 0 & -n_3 & n_2 & \upsilon_1 \\ n_3 & 0 & -n_1 & \upsilon_2 \\ -n_2 & n_1 & 0 & \upsilon_3 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.15}$$

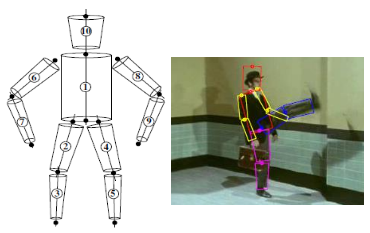
and the rigid body transformation becomes,

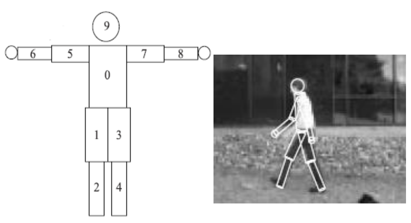
$$\exp(\widehat{\xi}\theta) = \begin{bmatrix} \exp(\widehat{n}\theta) & (I - \exp(\widehat{n}\theta)(n \times \upsilon) + nn^T\upsilon\theta \\ 0 & 1 \end{bmatrix}, \tag{2.16}$$

which is the element of $SE(3)$ group and $\exp(\widehat{n}\theta)$ can be computed through the *Rodrigues' formula* (Equation 2.14). For further reading, we refer [Sho85, MSZ94].

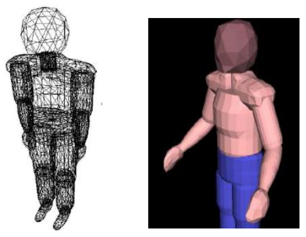
2.1.2 Joint Positions

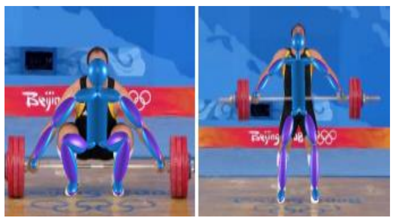
A human pose can be represented by the collection of the joint positions denoted as $\mathbf{X}$—the x, y, and z axes coordinates of the joints of the human body skeleton. The benefit to represent the human pose by the joint positions is that we can work directly in Euclidean space, and the disadvantage is that we may loss the rigid bone lengths as well as other anthropomorphic properties of the articulated human pose, which does not happen in case of the pose represented by the joint angle configurations. The pose represented by the joint positions is used especially in discriminative methods.


(a) Cylindrical model I


(b) Cylindrical model II

(c) Polymesh model


(d) Super-quadrics model


(e) Ellipsoids model

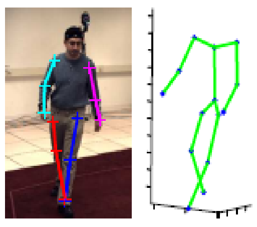

(f) Stick figure model

Figure 2.1: Human body models: **(a)** Cylindrical based body model I [SB06b] **(b)** Cylindrical based human body model II [SBF00] **(c)** Polymesh based model [SBB07] **(d)** Super-quadrics based model [GWK05] **(e)** Ellipsoids based human body model [SBB07] **(f)** Stick figure based model [WWL+14].

2.1.3 Body Parts

In this pose representation, the pose is parameterized by the different body parts. Each body part is considered as an individual rigid shape that is connected with the other body parts through a joint in order to structure the kinematic tree. Such types of pose representations are successfully used in human tracking as well as for the articulated human pose estimation. For segment representation, mostly the geometric primitives like cylinders, ellipsoids or truncated cones are employed [SB06b, RSB04, SBF00, ST01, BM98].

2.2 Human Body Model

The creation of a realistic representation of the human body model that elaborates the main characteristics of the human body is considered as a crucial step in 3D pose estimation. There are a variety of human body models found in literature, but the *stick figure* human body model [PMFR14, WWL+14, LC14, SSQTMN13, SSRA+12] is frequently used because it reflects the main characteristics of the human shape that should be considered during reconstruction of the 3D human pose. There are other approaches as well that utilize different human body models like some approaches utilize the geometry primitives—[SB06b, RSB04, SBF00] use cylinders and [ST01, BM98] deploy ellipsoids to develop the human body model. The geometric primitives need to be parameterized with the body segments' lengths, the scaling factors and their volumes *etc.*, manually or have to be estimated during calibration prior to tracking or pose estimation. More complex parametric shapes like super-quadrics have been presented in [GWK05, ST03b]. [SBB07] exploits a low-dimensional mesh based human body shape model with 25,000 polygons. A few examples of the human body models are illusterated in Figure 2.1.

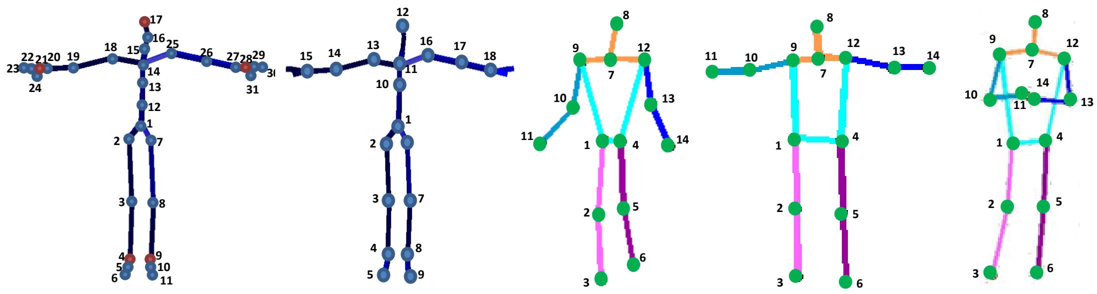
In general, the human body model consists of a variety of joints with different degree of freedom like: the ball-and-socket joints with three degrees of freedom *e.g.* the upper leg is connected to the hip; the hinge joints with one degree of freedom *e.g.* the knee joints; the saddle joints with two degrees of freedom *e.g.* like the ankle joints; the ellipsoidal joints with two degrees of freedom *e.g.* the wrist joints; and the plane joints with two degrees of freedom *e.g.* the clavicular joint in the shoulder.

The human body models also vary in number of joints/connectors that has been utilized to connect different body parts in order to represent the full body human pose. [WWL+14, LC14, SSQTMN13, SSRA+12] exploit 14 joints to represent the human full body pose, [AT06a] deploys the joint coordinates to represent human upper body with 24 dimensional pose vector. [FZZW14, RKS12, AB15] use 18 joints while [YKC13] utilize 15 joints for the full body human pose. [UFHF05] model the articulated human body structure with 84 degrees of freedom for walking action and 72 for golf action. [SB06b] uses 10 cylindrical objects to represent the body segments while [SBF00] employ 9 body segments in their tracking framework.

In our case, we use the *stick figure* body model throughout this dissertation and the body (skeleton) model varies in numbers of joints for different experimental and input scenarios. For example, for video-based 3D reconstruction in Chapters 4–5, we employ skeleton model with 31 joints. For image-based reconstruction, we utilize the body model containing 18 joints (Chapter 6) and the body model with 14 joints (Chapters 7), following the protocols of the state-of-the-art methods for the fair comparisons. All these skeleton models have been shown in Figure 2.2.

2.3 2D Human Pose

The image-based cues that are deployed in order to extract the 2D human pose from a video or a single image, can be categorised into two classes, the features-based models and the part-based models. We discuss both models one by one as below,



(a) HDM05 **(b)** CMU **(c)** HumanEva-I **(d)** Human3.6M **(e)** CMU

Figure 2.2: Human body models with joint specifications which are exploited in this dissertation: **(a)** HDM05 MoCap dataset with 31 joints used for the video-based analysis **(b)** CMU MoCap dataset with 18 joints used in Chapter 6 **(c)** HumanEva-I MoCap dataset with 14 joints **(d)** Human3.6M MoCap dataset with 14 joints **(e)** CMU MoCap dataset with 14 joints. **(c)**–**(e)** are utilized in image-based pose estimation in Chapter 7.

2.3.1 Features-based Models

There are a number of features that have been used to detect and extract the relevant interest points from the images like silhouettes (contours), edges, and the low-level features *i.e.* Scale-Invariant Feature Transform (SIFT), Saliency Scale (SS), Histograms of Oriented Gradients (HOG), Speeded-Up Robust Features (SURF) and Maximally Stable Extremal Regions (MSER) *etc.*

The most common image cues used for tracking and the pose estimation include silhouettes (contours), edges or combination of both [DR05, ST03a, SMH06]. Silhouettes and edges require a low computational cost and are invariant to illumination, clothing and considered well for the localization of a person. For the edge detection in an image, first the image is blurred and then convolved through Sobel filters which is specialized to detect lines. Silhouettes that are extracted from the multiple views are also utilized to reconstruct the 3D pose [RSH$^+$05]. The silhouettes-based features are inherently considered as ambiguous because multiple 3D human poses may have same 2D image based silhouette.

As our tracking approach is based on the low-level features, so we elaborate the low-level features in more details. An exemplary illustration of these low-level features has been shown in Figure 2.3. Among the low level features, the most popular is the Scale-Invariant Feature Transform (SIFT) features [Low04]. SIFT find extrema (local maxima/minima) in the scale-space through computing the Difference-of-Gaussian (DoG) image pyramid. The Difference-of-Gaussian is the approximation of the Laplacian-of-Gaussian (LoG) and invariant to scale and rotation. In second step of key points localization, the certain candidates for the interest points are chosen—which have low contrast and are poorly localized along the edges. This is done through Taylor series expansion of DoG, the principal curvatures and the Hessian matrix at the keypoint location. The weighted direction histogram is generated in a neighborhood of a keypoint with 36 bins. The third step of the orientation assignment has been performed by computing the gradient magnitude and the orientation in a region all around the keypoint. In the final step of the key point descriptor, these samples has been collected in 4×4 subregions with 8 bins. SIFT is invariant to translation, scaling and orientation as well as robust to illumination changes and affine distortions.

In Saliency Scale (SS) features [KB01], the authors do not take into consideration the image derivatives but in contrast rely on the salient regions. The saliency of the region is measured by computing the entropy of the probability distribution function of the intensity in a circular neighbourhood of that region. The authors look for the salient features with maximum entropy. In scale space, those scales are selected where the entropy is at its peak and is weighted according to the feature's self-dissimilarity. The Saliency Scale features are invariant to scale and view point changes.

[DT05] introduces Histograms of Oriented Gradients (HOG) features for human detection. They convolve a template of HOG descriptors at different image scales and then uses a Support Vector Machine (SVM) to discriminate the persons from other image structures. They compute the centered horizontal and vertical gradients with simple 1-dimensional mask $[-1\ 0\ 1]$ at $\sigma = 0$ and then compute the gradient orientations and magnitudes. They divide the image into 16×16 blocks with 50% overlap and quantize the gradient orientation into 9 bins with spacing between 0–180 degrees. The gradient with highest magnitude has been selected. At the end, these gradient histograms are collected in order to develop the feature vectors.

The Speeded-Up Robust Features (SURF) [BETVG08] are based on the same techniques and principles as in SIFT. For interest point detection, the authors compute fast Hessian matrix and detect blobs with maximum determinant. The scale space is developed by up-scaling the filter size in contrast to SIFT principles where the images are down scaled to build a pyramid. The scale space starts with the filter of size 9×9, later on the size of the filter increases with 15×15, 21×21 and 27×27. For a new octave, the gap between filter sizes has been doubled. The convolution is approximated with the integral images which make the whole process fast. The interest points are localized through the interpolation of maxima of the Hessian matrix determinant in the scale space as well as in the image space. The Haar wavelet is used in order to assign orientation to the interest point. For that purpose, the Haar wavelet responses in x and y-directions have been calculated in a circular region and to speed up the process again the integral images are used. Both responses are summed up to create the orientation vector and the longest vector assign the orientation to the interest point. The keypoint

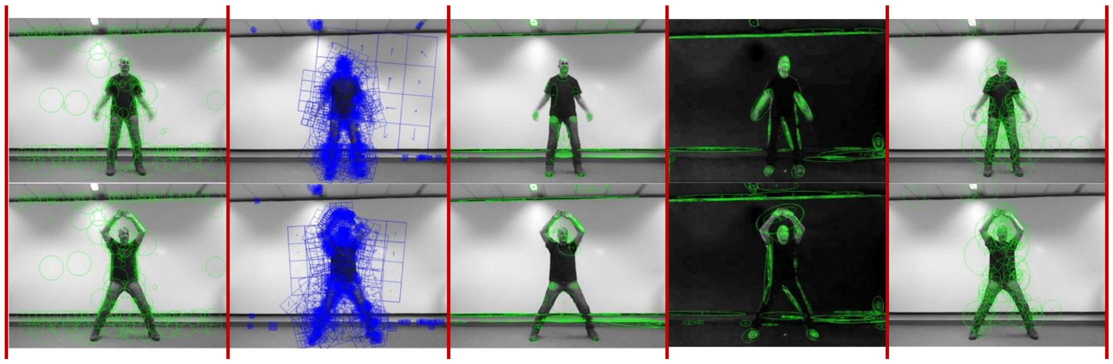


Figure 2.3: The low-level image-based features: **(a)** Scale Saliency features **(b)** Scale-Invariant Feature Transform **(c)** Maximally Stable Extremal Regions **(d)** color Maximally Stable Extremal Regions **(e)** Speeded-Up Robust Features.

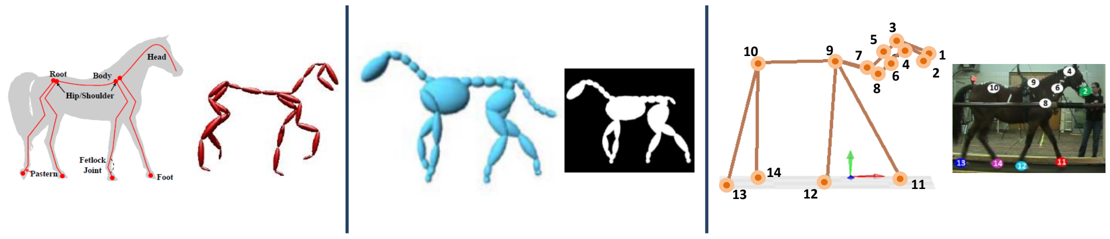
descriptor is based on Haar wavelet responses. The region splits up into 4×4 subregions and each subregion has a four dimensional vector for its intensity structure. As a result, the descriptor vector becomes $4 \times 4 \times 4 = 64$ dimensional vector whereas the SIFT descriptor vector is of $4 \times 4 \times 8 = 128$ length. For further reading, we refer [BETVG08].

The Maximally Stable Extremal Regions (MSER) [MCUP02] is a blob detector that finds interest points with maximum intensity region or minimum intensity region on the basis of binary intensity thresholds. It is a stable connected component of a set of binary images created through different levels of thresholds. A region is considered stable that remains consistent against intensity and does not vary in size. The maximal stable region with corresponding threshold is recorded for MSER descriptor and is approximated with an ellipse shape. MSER is invariant to affine transformations such as if the image is warped or skewed.

The color-based extension of Maximally Stable Extremal Regions (colorMSER) is proposed by [CG11], where he authors shift the MSER algorithm in HSV (Hue, Saturation, Value) color space. For an example, with the given hue h, saturation s and the value v, the *red* channel is characterized as $red(h, s, v) = |180–h| * s * v$. The similar functions are applied for the *green* and *blue* channels. As a result, the good intensity discriminations has been found. The extended interest regions are more robust to illumination, the backgrounds and the image noise.

2.3.2 Part-based Models

The first part-based model “Pictorial Structures” was presented by [FE73] in order to detect the face, which exploits the appearance of the objects as a set of connected parts. For full body human pose estimation, the classic part-based approach is to develop



(a) [HHL13] **(b)** [FRDC04] **(c)** The presented model

Figure 2.4: Figures **(a)**–**(b)** show ellipsoid based horse body models while **(c)** demonstrates the *stick figure* horse body model which we deploy in order to reconstruct 3D horse motions in Chapter 8.

human model based on a set of body parts such as, a head, a torso, the arms and the legs. Such part-based models are often demonstrated through a graph $\mathcal{G} = (\mathcal{J}, \mathcal{L})$, where the vertices $\mathcal{J}$ corresponds to the 2D locations of $\mathcal{N}_\mathrm{j}$ number of parts (joints) and the edges $\mathcal{L}$ correspond to the kinematic constraints between them [FH05, FMR08, YR11, DLGVG14]. To infer the 2D pose, commonly there are two important components: (i) how accurately each part matches the image feature vector extracted at its location, and (ii) how accurately the relative locations of the parts can be computed through deformable model. More precisely, the first, the unary potential, scores a local match for placing a part template through local features vector extracted at that location, and the second, the binary potentials, compute the deformation cost through evaluating the relative locations of the pairs of the parts. In this dissertation, we benefit from the part-based models described in [YR11] and [DLGVG14]. In [YR11] the mixture of HOG features are utilized to detect different human body parts and utilize tree structure with $\mathcal{N}_\mathrm{j} = 14$ and $\mathcal{N}_\mathrm{j} = 26$ number of nodes (parts) for co-occurrence and spatial relationships between parts. The authors use Support Vector Machine (SVM) for discriminative learning. We employ [YR11] off-the-shelf algorithm to detect 2D pose from an image in Chapter 6. In [DLGVG14] the authors utilize HOG features with 9 orientation bins with a cell of size 5×5. They further apply max-filtration by a 5×5 kernel in order to maximize the HOG-filter responses to the neighboring pixels. They introduce body parts dependent random forests as a discriminative model. We follow [DLGVG14] and choose random forest based joint regressors with 14 joints (Chapters 7).

2.4 Quadruped Models

For quadruped motions, [HHL13, FRDC04] use ellipsoid based horse body model to recover 3D pose from horse images. In [HHL13] the horse body model consists of 35 joints with 61 degrees of freedom. They set spine of the horse as a root of the body with 6 degrees of freedom (3 for global orientation and 3 for position). Their proposed body model has been shown in Figure 2.4 (a). In [FRDC04] the 3D motions are estimated from the extracted regions of interest as well as the silhouettes from video input by the standard morphological analysis after background subtraction. They do not depend on joints' labelling, instead rely on the raw labeling of joints extracted from the main features of the spine, the legs, and the head. They apply PCA on the binary images and interpolate the optimal examples of the 3D poses. In case of failure in segmentation, they use sketching tool to label the pose in the video.

The quadrupedal body structure that we employ in this dissertation (Chapter 8) consists of 15 joints (all available 14 joints with one virtual marker) in order to reconstruct the 3D poses for the quadruped motions (Figure 2.4 (c)). Unfortunately, we do not have joint angles parameterizations in the MoCap dataset, that's why we exploit the joint positions in Euclidean space directly to estimate the 3D quadruped motions.

3 Related Work

Student: Dr. Einstein, Aren't these the same questions as last year's final exam?
Dr. Einstein: Yes; But this year the answers are different.

Albert Einstein

In this chapter, we discuss briefly the existing approaches for video or images-based 3D retrieval and reconstruction of human poses.

3.1 Generative and Discriminative Approaches

A variety of approaches has been investigated in order to estimate 3D human poses. These approaches can be classified into three types of categories such as discriminative approaches, generative approaches and the hybrid approaches.

- **Discriminative Approaches.** Earlier approaches for monocular 3D human pose estimation [BS10, AT04, SKLM05, AT06b, BSKM08, MM06, YKC13] utilize discriminative methods to learn a mapping from local image features (*e.g.* HOG, SIFT, *etc.*) to 3D human pose or use a deep convolutional neural network [LC14, LZC15]. Since the local features are sensitive to noise, these methods often assume that the location and scale of the human is given, *e.g.*, in the form of an accurate bounding box or silhouette. Such types of discriminative approaches need a large amount of training dataset to deal with a variety of viewpoints. While the recent approach [ICS14] still relies on the known silhouette of the human body, it partially overcomes the limitations of local image features by segmenting the body parts and using a second order hierarchical pooling process to obtain robust descriptors. These approaches are fast but sensitive to noise and do not generalize well for poor image-based evidences [SB06a].
- **Generative Approaches.** The generative approaches [DBR00, UFHF05, UFF06, CC09, PMBG^{+}11, AB15] are model-based approaches and commonly model on

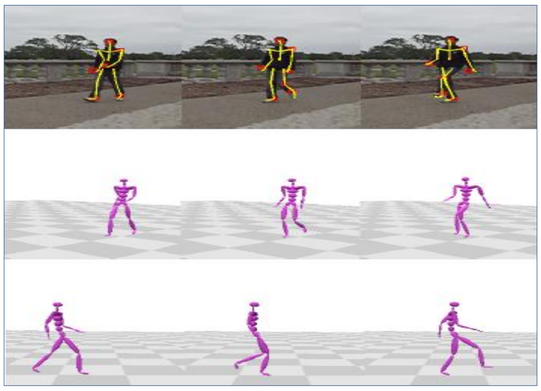
randomly generating observable data and then rely on the best alignment. The generative approaches formulate some energy function that optimizes the model to fit according to the observations. The difficulties with generative approaches are: (i) they usually need some realistic good initialization, and (ii) there may exist multiple local models. The multiple models' ambiguities are addressed through activity specific models [SU10a, WFH08]. Despite of these difficulties, the generative approaches typically do not sensitive to noise and can deal adequately even with poor image evidences [SB06a].

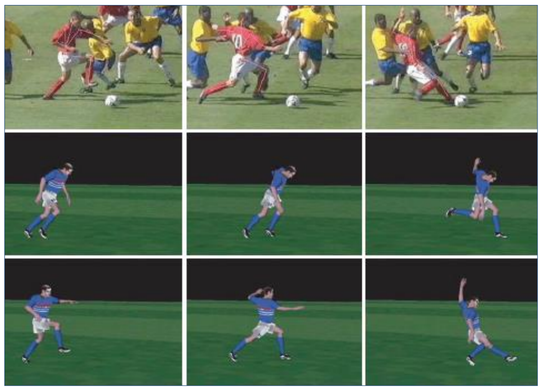
- **Hybrid Approaches.** There are some hybrid approaches which combine generative and discriminative methods together such as [SSQTMN13, KG14, PMFR14]. In [SSQTMN13], the authors first predict the 3D human pose hypotheses by combining probabilistic generative kinematic models and then utilize HOG features based discriminative 2D part detectors which weight these hypotheses. The whole process is repeated until the method converges. The 3D Pictorial Structure Model (PSM) proposed in [KG14] combines generative and discriminative methods. Regression forests are trained to estimate the probabilities of 3D joint locations and the final 3D pose is inferred by the PSM. Since inference is performed in 3D, the bounding volume of the 3D pose space needs to be known and the inference requires a few minutes per frame.

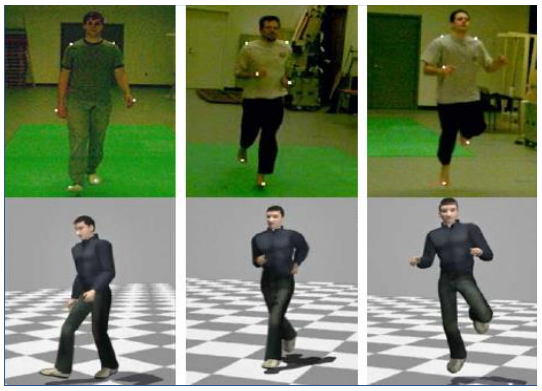
Besides of a-priori knowledge about bounding volumes, bounding boxes or silhouettes, these approaches require sufficient training images with annotated 3D poses. Since such training data is very difficult to acquire, we propose approaches where we do not need corresponding 3D-2D pairs, which is not only costly but also need studio-like hardware setup and calibration. Our proposed frameworks do not require training images with 3D annotations, but exploits existing motion capture datasets to estimate the 3D human pose. In short, we integrate both capturing approaches, the 3D marker-based motion capturing and a simple RGB camera based capturing of poses, together in order to estimate the final 3D poses.

3.2 Input-based Classifications

Some general methods to reconstruct the human poses found in prior literature can be categorized on the basis of 2D inputs such as images, videos, sketches and kinect-based inputs. With any type of input query and in any case, generative or discriminative, these approaches work with 2D image-based cues and features that may be detected or tracked in the form of joint positions, silhouettes, contours or edges. We classify


(a) Chen and Chai [CC09].


(b) Park et al. [PCS02].


(c) Chai and Hodgins [CH05].

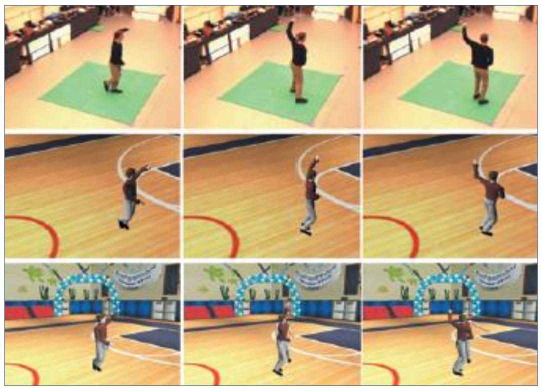

(d) Ren *et al.* [RSH+05].

Figure 3.1: A few examples for video-based 3D motion reconstruction that exploit the motion capture library as a prior existing knowledge.

these approaches on the basis of 2D inputs as well as on the local image-based extracted cues/features that are given to the system for final 3D pose estimation.

3.2.1 Video-based 3D Motion Reconstruction

A number of solutions has been purposed for the 3D human motion reconstruction from the video input, either a monocular video or the video from multi-camera system. We describe an overview of few approaches as under.

Some approaches construct statistical human pose models like transforming 2D silhouettes and contours into 3D human poses and motions [RSH$^+$05, SKLM05, AT04, EsL04, ST02, LCR$^+$02]. Sminchisescu and Telea [ST02] extract the 2D human silhouettes, predict articulation as well as the structural key parameters of the human model and fit it onto the image-based observations through an energy minimization procedure. In [AT04] the 3D human poses are recovered directly from the image sequences through a sparse Bayesian nonlinear regression without exploiting any parametric structural human model. Ren *et al.* [RSH$^+$05] propose a method for animating the 3D human motion, which exploits silhouettes extracted from three video cameras and a MoCap library in order to select those features which are close enough to estimate the orientation of the human character as well as to recover the 3D human body configuration. Roodsarabi and Behrad [RB08] describe an approach for the 3D human motion reconstruction from the video by employing Taylor method and utilize Discrete Cosine Transform (DCT) as descriptors in the process of matching and tracking.

A few approaches propose the action specific priors in order to track the human and estimate the 3D pose jointly [UFF06, ARS10, YGVG12, YKC13]. In [ARS10] the pedestrian are first detected and then coupled it with deformable body parts in order to recover the final 3D human poses in an outdoor environment. Yao *et al.* [YGVG12] utilize the 2D appearance-based action specific priors for the 3D human pose reconstruction. Yu *et al.* [YKC13] model the spatiotemporal action priors integrated with part-based 2D pose prediction for estimation of the 3D human poses. Despite of these approaches, action specific priors are still insufficient for 3D human pose prediction due to the reason that there may exist multiple poses recognized for the same one action category which leads to create ambiguities in pose estimation process.

A number of approaches for 3D reconstruction of human motions found in the literature that exploit geometric constraints [PS11, RSB+08] and the physics-based modeling coupled with image-based features [WC10, VSJ08]. Wei and Chai in [WC10] reconstruct the 3D human motion from a monocular input video through the physics-based modeling and the rigid body constraints. They combine physical constraints with image based observations to estimate the human skeleton, the camera parameters and the final 3D human poses for a few key frames from the video. Their system require a minimal user interaction to annotate the key intermediate frames for tracking. In fact, they model 3D human poses for the interactive intermediate key frames and then predict in-between 3D poses by interpolation of these key frames. They impose rigid body parameterizations with bone projection and bone symmetry constraints to compute the human skeleton.

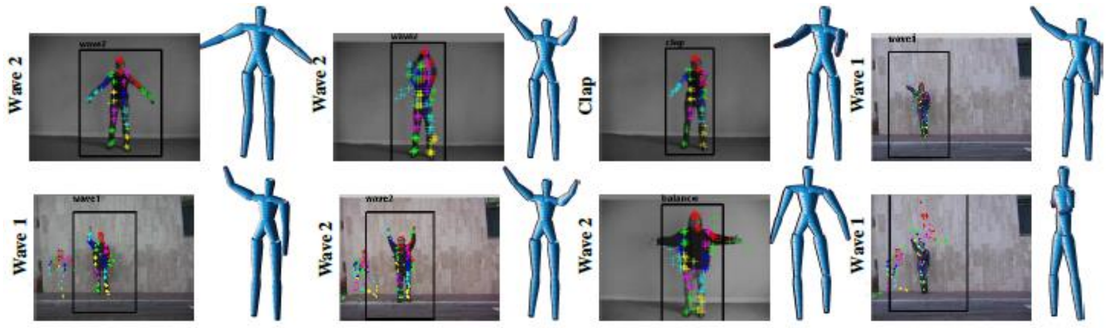
There are many methods which exploit some prior knowledge from motion capture dataset for the 3D reconstruction [PCS02, RSB+08, CC09, CH05, RSH+05]. Chen and Chai [CC09] propose a model-based pipeline to reconstruct the 3D human motion sequences from an uncalibrated monocular video. They employ just a small set of 2D joint trajectories tracked from a monocular input video and learn a 3D pose model by applying Principal Component Analysis with weighted combination of eigenvectors and map it with 2D image features by nonlinear gradient based optimization framework. They also estimate the human skeleton and the camera parameters. They enforce the absolute length of the segments of the skeleton with 24 bones, and thus keep the segment proportion constant in the 3D reconstruction. Park *et al.* [PCS02] describe a novel method for the human motion reconstruction from the inter-frame feature correspondence in the video streaming by employing the MoCap library. They first make the reference motion to fit onto the video through time warping and then find out the orientations of the joints & root trajectories in order to predict the 3D human motion. The prior knowledge from MoCap library is sometime embedded into implementation of some constraints only like Rosenhahn *et al.* [RSB+08] utilize geometric prior information about the movement pattern in the process of markerless pose tracking. Chai and Hodgins [CH05] reconstruct 3D human motions in constrained environment by employing a MoCap library, two synchronized cameras and a small set (6-9) of retro-reflective markers. For the accurate 2D pose extraction from the video, they apply background subtraction, color similarity constraints, impose specific illumination environment and utilize epipolar geometry. They introduce *neighbor graph* to search into MoCap library for the relevant nearest neighbor which are further used in motion reconstruction process.

Many of the aforementioned approaches necessitate an accurate extraction of 2D human pose features or dependent on the special setups *i.e.* two/three synchronized camera system, retro-reflective markers or the controlled environments to estimate 3D human poses. In contrast, our video-based generative pipeline needs no any costly hardware setup and work with the monocular video sequences without any pre-assumption. Furthermore, our methodology utilize just five 2D joint locations (the four end effectors and the head) to search into MoCap database for similar poses which are exploited in order to reconstruct the final 3D human poses.

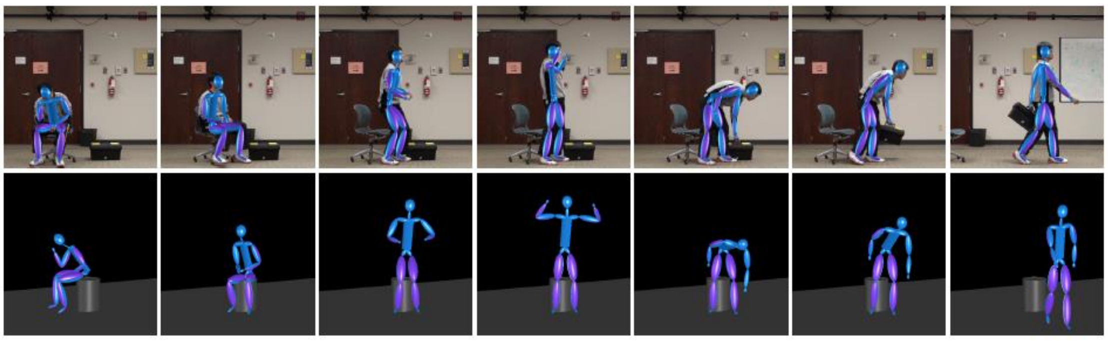
3.2.2 Image-based 3D Pose Estimation

Several approaches utilize 2D joint locations of the human body extracted from a single monocular image to predict the 3D human pose. In some methodlogies [VL10, WC09, HDK07, ZLCK05], the 2D joint locations in a image have been manually labelled, a few approaches [FZZW14, RKS12, AB15] work on synthetic 2D input images, some methods [SSRA+12, WWL+14, RDG13] make use of off-the-shelf 2D detector to estimate the 2D pose from an image, and just a few approaches [SSQTMN13, LC14] estimate the 2D human pose and the 3D human poses together.

- **Multiple Images.** A common approach for 3D human pose estimation is to utilize multiple images captured by synchronized cameras [BAA+14, SIHB12, YGVG12] or multiple synthetic images [WC09, VL10]. Wei and Chai [WC09] estimate the 3D human poses, the weak perspective camera parameters and the human skeleton from the 2D point correspondences. They also predict 3D poses from manually labelled 2D monocular images. They impose bone projection constraints using weak perspective camera model and symmetric properties of the bone segments. They introduce a new set of rigid body constraints by computing distances *i.e.*: between



(a) Yu *et al.* [YKC13].



(b) Wei and Chai [WC10].

Figure 3.2: Examples for video-based 3D motion reconstruction; **(a)** represents action priors integrated with part-based 2D features in order to predict 3D pose, whereas **(b)** shows physics-based modeling coupled with image-based features to reconstruct the final 3D motion.

hip joints; between left and right shoulder joints; root and left shoulder joints; root and right shoulder joint *etc.* They conclude in their work that their system needs a few single view images *i.e.* ≥ 5 to predict symmetric human skeletal. The authors [VL10] recover the 3D pose from an uncalibrated 2D point correspondences through least-square method combined with the assumption about rigid human body constraints. They argue on Wei and Chai's method [WC09] and claim that rigid body constraints exist for only few body sub-structures, not for the entire human body. The requirement of a multi-camera system in a controlled environment, however, limits the applicability of these methods.

- **Depth Images.** Since 3D human pose estimation from an image is very challenging due to missing depth information, as a solution the depth images have been utilized for human pose estimation [BMB^{+}11, SFC^{+}11, GWK05]. However, current depth sensors are limited to indoor environments and cannot be used in unconstrained scenarios.

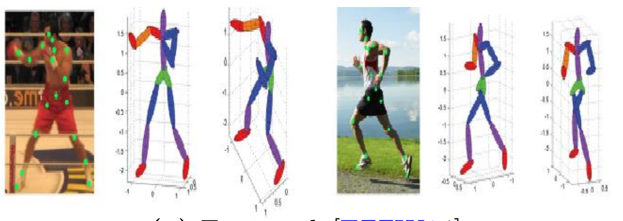
- **Single Image.** Estimating 3D human pose from a 2D pose extracted from a single image by exploiting motion capture data has been addressed in a few works [SSRA+12, RKS12, SSQTMN13, WWL+14, FZZW14, AB15]. These data-driven approaches for 3D pose estimation require some sophisticated dimensionality reduction method in order to represent a solution space. A common method is the formation of low-dimensional local model learned from some prior existing knowledge [SHP04, CH05, BSB+07, JSMH12].

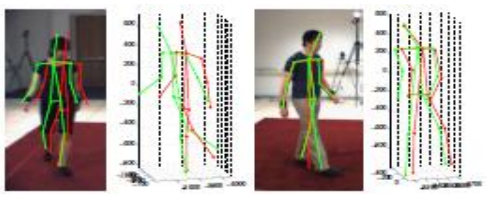
Manually Labelled Inputs. Hornung *et al.* [HDK07] animate 2D pictures with the help of user interaction in the form of selection of joints and the prior information available in 3D motion capture dataset. They present a shape deformation method in order to animate the still image projectively. [ZLCK05] reconstruct the 3D human pose from manually labelled 2D feature points of a monocular image through imposing biomechanical constraints and the Genetic Algorithm for optimization process.

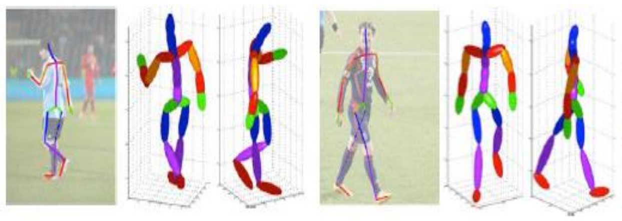
Instead of predicting poses with a low 3D joint localization error, an approach for retrieving semantic meaningful poses is proposed in [PMFR14].

Synthetic Inputs. Ramakrishna *et al.* [RKS12] propose an activity-independent approach where they develop an over-complete dictionary of basis vectors by concatenating bases from different action categories for 3D pose modeling using the MoCap dataset and fit the model to manually annotated 2D joint locations. Their model also has a sparse set of basis vectors which they estimate using Orthogonal Matching Pursuit (OMP). They emphasize weak anthropometric constraints in the form of summation of limbs lengths. Fan *et al.* [FZZW14] enhances Ramakrishna approach [RKS12] and propose Pose Locality Constrained Representation (PLCR) based on a large amount of 3D pose training data for 3D human pose estimation. They first develop a hierarchical pose tree on the basis of clustering and sub-clustering of pose data. They construct a block-structural pose dictionary which is based on all the subspaces in the pose tree.

Off-the-shelf Estimator based Inputs. Wang *et al.* [WWL+14] estimate 3D pose by using a set of basis vectors with addition of sparsity and anthropometric constraints. They handle 2D poses estimation using an off-the-shelf 2D pose estimator [YR11]. They minimize their objective function using L_1 norm error. The same 2D pose estimator is also used in [SSRA+12, SSQTMN13] to constrain the search space of 3D poses. In [SSRA+12] an evolutionary algorithm is used to sample poses from the pose space that correspond to the estimated 2D joint positions. This set is then exhaustively evaluated according to some anthropometric constraints. They model a 3D pose with linear combinations of a number of deformation modes obtained through Principal Component Analysis. They represent each joint position with Gaussian distribution which is forwarded to a set of ambiguous 3D shapes


(a) Fan *et al.* [FZZW14].


(b) Simo-Serra *et al.* [SSQTMN13].


(c) Ramakrishna *et al.* [RKS12].

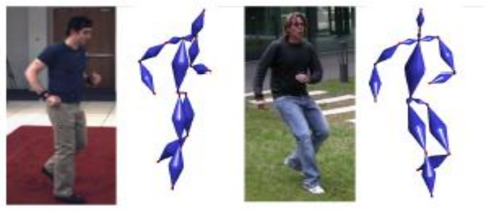

(d) Pons-Moll *et al.* [PMFR14].

Figure 3.3: A few examples for image-based 3D pose reconstruction where a single image is given as an input.

and then one-class Support Vector Machine (OCSVM) is utilized to find out the deviation of the 3D pose from the training set.

The approach [SSRA+12] is extended in [SSQTMN13] such that the 2D pose estimation and 3D pose estimation are iterated. The authors in [LC14] use a deep Convolution Neural Network (CNN) for 3D human pose estimation. The authors train the pose regression as well as the body part detection together by developing a shared network where the gradients can back-propagated and get benefits from shared features of both tasks. In contrast to [RKS12, WWL+14, SSRA+12], [SSQTMN13] deals with 2D pose estimation errors. Our approach also estimates 2D and 3D pose iteratively but it is faster and more accurate than the sampling based approach [SSQTMN13]. We design multiple feature set to retrieve robust nearest neighbors from the MoCap dataset, which are exploited further in reconstruction process with multiple energy terms in order to recover the final 3D pose.

3.2.3 Sketches based 3D Pose Reconstruction

Another class of research which predict the 3D human pose from the 2D hand-drawn stick figures [DAC+03, MQW07, JSMH12, CYI+12, YVN+14]. A sketching interface has been presented in [DAC+03], where the drawing skill of the artist has been coupled with the pose reconstruction algorithm iteratively in order to reconstruct the robust 3D pose from the hand-drawn sketch. Jain *et al.* [JSMH12] reconstruct the 3D animation from the 2D hand-drawn characters and create an interaction between the 3D reconstructed character and a virtual world. The user-defined orthographic camera model is used which is optimized through geometric projection error. The authors minimize the skeleton ambiguities through enforcing the hand-drawn animation limbs' length to match with the actual limbs' length of the skeleton computed from pre-existing knowledge in the MoCap dataset. In [CYI+12] the authors first conduct a survey and user study about how people draw sketches and then propose a method in order to convert the motion capture data into the stick figures. With an input sketch, the search and retrieval is performed where the 2D stick figures act as a medium for visualizing and searching into MoCap data. Yoo *et al.* [YVN+14] introduce a pipeline for fast animations of the 3D human characters from the 2D sketches using the MoCap library. The authors use the rotation curve cues to improve the 2D sketches and as a result the searching procedure and the final 3D human character animation become more accurate and robust.

3.2.4 Kinect based 3D Pose Reconstruction

A few approaches recover the 3D pose from an incomplete Kinect based captured human pose [SH12, ZLLS14]. In [SH12], authors propose a framework in order to synthesize a full human posture from an incomplete pose captured through Microsoft Kinect camera. They search for the best similar poses into the motion capture dataset and refine the pose estimation with the positional and rotational controlling constraints. Additionally, they also use the environmental constraints like external forces and torques computed by a PD controller. Zhou *et al.* [ZLLS14] adopt a non-parametric Gaussian process model as a prior in order to leverage the joints' position obtained from Microsoft Kinect and the motion capture dataset. With the learned Gaussian model, they predict the offsets between the MoCap pose and the Kinect pose as a conditional probability distribution and thus yield the spatial prediction energy term. They also use temporal information to avoids velocity variations and to enforce smoothness between successive frames. The authors further impose joint's reliability in three aspects such as behavior reliability, kinematics reliability, and tracking state reliability.

3.3 2D Image Features for 3D Reconstruction

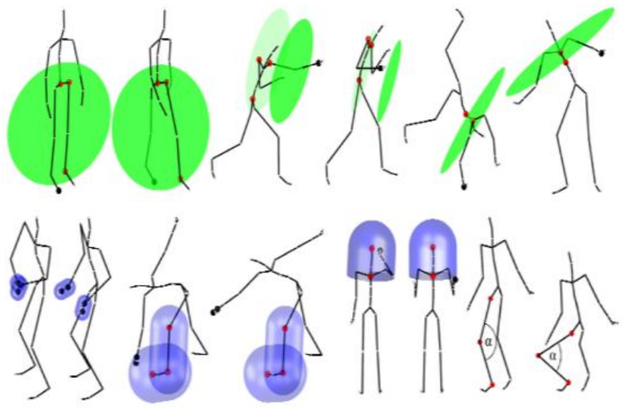
A lot of existing approaches for 2D pose detection and tracking found in the literature mainly focus on 2D image cues based on some appearance evidences like color, shape, edges, distribution and/or knowledge of background, silhouette and contour [DR05, ST03a] as well as the low-level local features like: SIFT [Low04], Saliency Scale [KB01], HOG features [DT05], SURF [BETVG08], MSER [MCUP02] and colorMSER [CG11] *etc.* A few approaches exploit the body parts kinematics combined with the image cues/features in order to detect the human pose in an image [FH05, FMR08, YR11, DLGVG14]. The details about these approaches have been discussed in Section 2.3.

Therefore, we focus here only those 2D pose estimation techniques which are employed to recover the final 3D human pose. The common image cues and features for automatic 2D pose estimation that are exploited further for 3D pose prediction include: shape contexts [SKLM05, AT06b]; silhouette and edges [DR05, ST03a, SMH06]; texture-based features [BMP04]; Histogram of Oriented Gradients (HOG) features [BS10, OS08, SVD03]; optical or motion flow [FB02, ST03a, VBK05]; Scale-Invariant Feature Transform (SIFT) descriptors [BSKM08, AT06a, KSM07, SLK11] and pictorial structure model [KG14, WWL+14, RDG13, SSRA+12] *etc.* The method [SU10b] utilize the image features like histograms of SIFT features and PHOG features to work on the rectified images.

The silhouettes-based features create ambiguity because more than two 3D human poses may have very similar 2D image based silhouette. Scale-Invariant Feature Transform (SIFT) descriptors are efficient but it is computationally very expensive. In our case, we employ SURF (Speeded Up Robust Features) which is an approximation of SIFT with significantly low computation—almost 6 time faster and more robust to image noise as compared to SIFT. We combine SURF with MSER (Maximally Stable Extremal Regions) and colorMSER features to detect and track 2D joints in video-based motion reconstruction. In image-based pose estimation, we employ pictorial structure model [YR11, DLGVG14] to detect 2D joint positions.

3.4 3D Pose Retrieval

The increasing amount of available motion capture data and the data-driven methods require to make use of efficient motion retrieval strategies. Kovar and Gleicher [KG04] propose *Match Webs* to index motion capture databases. This method has quadratic complexity with the size of the motion capture database, since a local distance matrix has to be computed in comparison with each pair of frames. The same complexity holds for the computation of a *neighbor graph*, the data structure presented by Chai and Hodgins [CH05]. Müller *et al.* [MRC05] introduce boolean features to segment human motion capture data. On combination with an efficient lookup based on inverted lists, they retrieve logical similar motion with a complexity of $\mathsf{N} \log \mathsf{N}$. The work of Krüger *et al.* [KTWZ10] presents a fast method to search for numerically similar poses and extends pose matching to motion matching by employing a so called *lazy neighborhood graph*. The authors compare feature sets of different dimensions and found that a 15-dimensional feature set based on the positions of hands, feet and head can describe human poses accurately and the best choice for fast similarity search. Later on, Tautges *et al.* [TZK+11] enhance the *lazy neighborhood graph* into incremental online version *online lazy neighbourhood graph* (OLNG) and reconstruct human motions using sparse accelerometer data. Wu *et al.* [WTR11] create adaptive clusters using k-mean


(a) Müller et al. [MRC05].

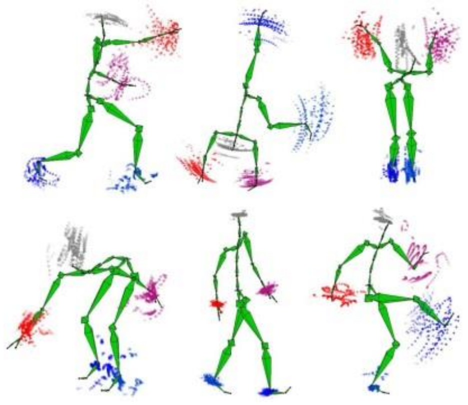

(b) Krüger et al. [KTWZ10].

Figure 3.4: A few examples which demonstrate the features that are used for retrieval of similar poses from the MoCap dataset.

clustering algorithm and then build kd-tree on the basis of these clustering centers in order to extract the character pose from a large motion capture database. Choensawat *et al.* [CCH12] propose a retrieval strategy from the MoCap dataset based on the joint speed as well as the variations in speed patterns for a short interval. In fact, they compute derivative of the joint speed and utilize it for similarity search.

3.5 3D Reconstruction of Quadruped Motion

An overview on the previous works in computer animation specific to the quadruped motions is given in the STAReport [SRH+08]. Most of the work regarding 3D reconstruction from a video or an image has been done on the human motions as mentioned in Section 3.2 and there also exists a few standard motion capture datasets for the human motions [CMU14, GFB12, MRC+07, IPOS14, SBB10] which are publically accessible too.

A very few work has been found in the literature with respect to reconstruction of quadruped species motion. Huang *et al.* [HHL13] synthesize horse motion sequences driven by the photographs of the horses. They manually annotate the horse motions in the Eadweard Muybridge's photographs through MAYA during preprocessing step. They also present asynchronous time warping strategy in order to adjust the horse's gait speed, direction and the transition between different motions. In [WV03], the authors propose 3D reconstruction method for the human and the animal. They make use of contour detection techniques and fit a 3D model onto the extracted 2D contours. For slow motions and simple backgrounds, this technique produces acceptable results but a considerable user interaction is needed for the sequences that contain more complex motions. The authors in [FRDC04] extract image parameters through applying Principal

Component Analysis on binary input key frames of the video, and employ them to recover 3D wild life motion sequences. Later on, they interpolate these key frames using Radial Basis Function (RBF). They also introduce the criterion on which basis the key frames from the video can be selected and utilized to generate the final 3D motion sequences. We are not aware of any data-driven method in this context, therefore, there is a vital need to record systematic quadruped motion capture databases.

Part I

Motion Reconstruction from Video

4

Motion Reconstruction from Video

Truth is ever to be found in simplicity, and not in the multiplicity and confusion of things.

Isaac Newton

4.1 Introduction

A lot of techniques have been proposed for human motion analysis and the 3D reconstruction of motions but one of the most widespread and successful approaches is a model-based generative approach. This chapter introduces a novel framework for the model-based full body human motion reconstruction from the 2D video data, using a motion capture dataset as a prior existing domain knowledge which contains a large amount of the 3D poses having complete knowledge about how people perform a variety of activities. Our proposed approach is based on the efficient search and retrieval of nearest neighbors from the motion capture (MoCap) dataset, which is the major and key component in almost all data-driven applications. One strength of our approach is that we need just only five joint locations (the 2D feature sets of the four end effectors and the head) in order to search into the MoCap dataset for the similar poses. We exploit these retrieved similar poses further in the process of 3D motion reconstruction.

We start by extracting the 2D feature sets from a video input which may be in the form of synthetic 2D video generated from some motion capture file through random camera parameters or some real monocular video. In case of real input video sequences, we deploy the local feature detectors and descriptors like SURF (Speeded Up Robust Features) [BETVG08] coupled with MSER (Maximally Stable Extremal Regions) [MCUP02, DB06] in order to detect and track five joint locations (the four end effectors and the head). After extracting the 2D feature sets from the given 2D input video, we search and retrieve the similar poses from the motion capture dataset. We address the correspondence between 3D-2D pose-image data by developing a *knowledge base*—an intermediate container that contains the normalized 3D pose space as well as the corresponding normalized 2D pose space. We derive the 2D normalized poses from the 3D normalized poses through orthographic projection at different view directions in the form of azimuth and elevation angles. For 3D pose reconstruction, we learn a local 3D pose model in low dimensional Principal Component Analysis (PCA) subspace on the basis of the similar poses retrieved from the MoCap dataset. Accurate modeling of priors in some low dimensional space is the fundamental technique in almost all 3D reconstruction methods. We further constrain the local 3D pose model with some prior energy terms as well as the projection alignment. A series of local pose models has been learnt and utilized in creating the final global model of pose-by-pose full body human motion reconstruction.

Our proposed reconstruction framework is inspired by the work of Chai and Hodgins [CH05] but there are few very distinctive differences such as: (i) Chai and Hodgins [CH05] use two video cameras and synchronize them through epipolar geometry but in our case we work with a monocular video which is a more sparse input. (ii) [CH05] also deploy a small set of retro-reflective markers with calibrated cameras which limits the method to indoor controlled environment with various constraints *i.e.* illumination, synchronization between multiple cameras, synchronization with retro-reflective markers and the view directions. Moreover, such systems are not only costly but also problematic and troublesome in terms of creating the setup and calibration. In contrast, we do not rely on any other special setups (like retro-reflective markers or Inertial Measurement Units (IMUs)). As a result, our proposed system is more flexible and can capture motion in any indoor/outdoor environment. (iii) [CH05] endorse assumption that most of the retro-reflective markers must be seen by at least one camera which do not allow user to move freely but our proposed pipeline need not such a assumption about the movements of the performing actor. (iv) Chai and Hodgins [CH05] employ all 18 joints of their skeleton model for searching the similar poses into the MoCap dataset. In contrast, we utilize 2D locations of the just five specific joints (the four end effectors and the head) for search and retrieval of nearest neighbors from the MoCap dataset which makes our system faster.

4.2 Overview

We propose a data-driven motion retrieval and reconstruction pipeline, where we search into MoCap dataset for 3D motion sequences that are very similar to the input control signals. We develop the feature sets based on the positions of the four end effectors and the head [KTWZ10] on which basis we create nearest neighbor search. This will be discussed in Section 4.3.1. For the input query motions, we consider two scenarios: the synthetic examples which we create from the motion capture sequences through some

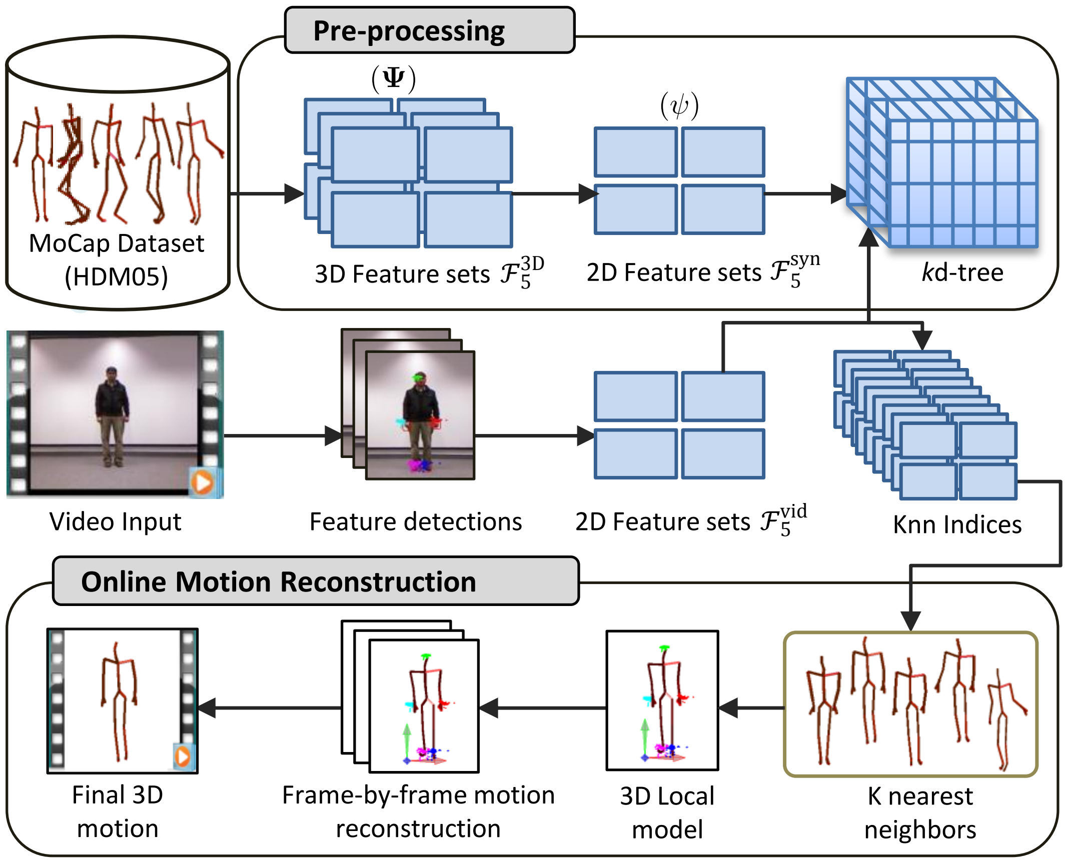

Pre-processing
MoCap Dataset (HDM05)
$(\mathbf{\Psi})$
3D Feature sets $\mathcal{F}_5^{3D}$
(ψ)
2D Feature sets $\mathcal{F}_5^{syn}$
*k*d-tree
Video Input
Feature detections
2D Feature sets $\mathcal{F}_5^{vid}$
Knn Indices
Online Motion Reconstruction
Final 3D motion
Frame-by-frame motion reconstruction
3D Local model
K nearest neighbors

Figure 4.1: System overview: In a pre-processing step, we develop a *knowledge base* consisting of the 3D and 2D normalized pose spaces, denoted as $(\mathbf{\Psi})$ and (ψ) respectively, using the MoCap library. The data structure *k*d-tree is built on the 2D normalized poses. The 2D feature sets extracted from the 2D input video sequences are given as input to the system for K nearest neighbor search. These retrieved closest poses are then exploited in learning the 3D local pose model as well as in the final frame-by-frame motion reconstruction process.

random camera parameters and the real video motion clips. In case of real video input, the relevant 2D features are detected and tracked using SURF and MSER feature detector/descriptors and are used as query for the similarity search. This part is described in Section 4.3.2. As we are dealing with the 2D inputs, therefore for the correspondence between 2D-3D data, the 3D feature sets extracted from the MoCap data are projected onto the 2D image-plane at different azimuth and elevation angles through an orthographic transformation and as a result we get the 10-dimensional feature sets which are used further to build a spatial data structure—in our case we use a *k*d-tree. With the feature sets in hand extracted from the query motion sequences, a K nearest neighbor (Knn) search is performed. This will be discussed in Section 4.3.3. We built a 3D local pose model that is constrained by the MoCap priors and is mapped onto input

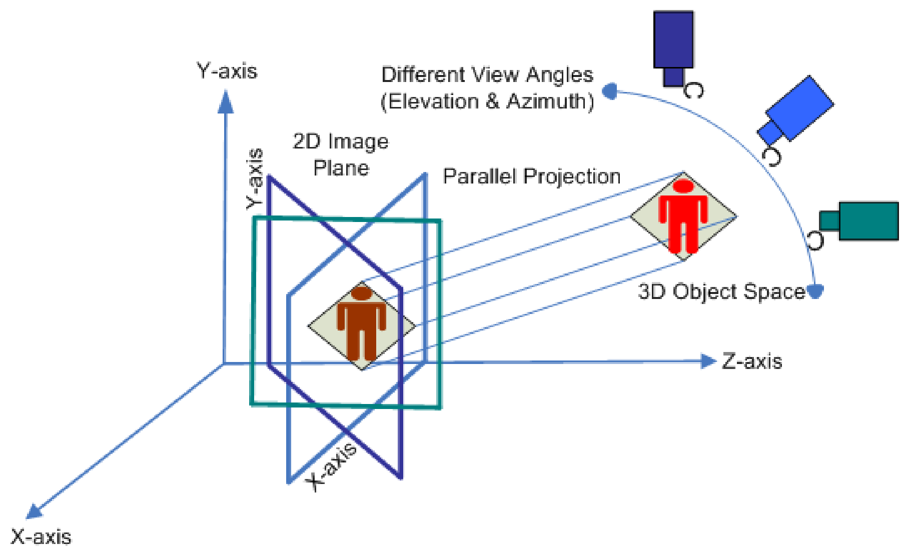
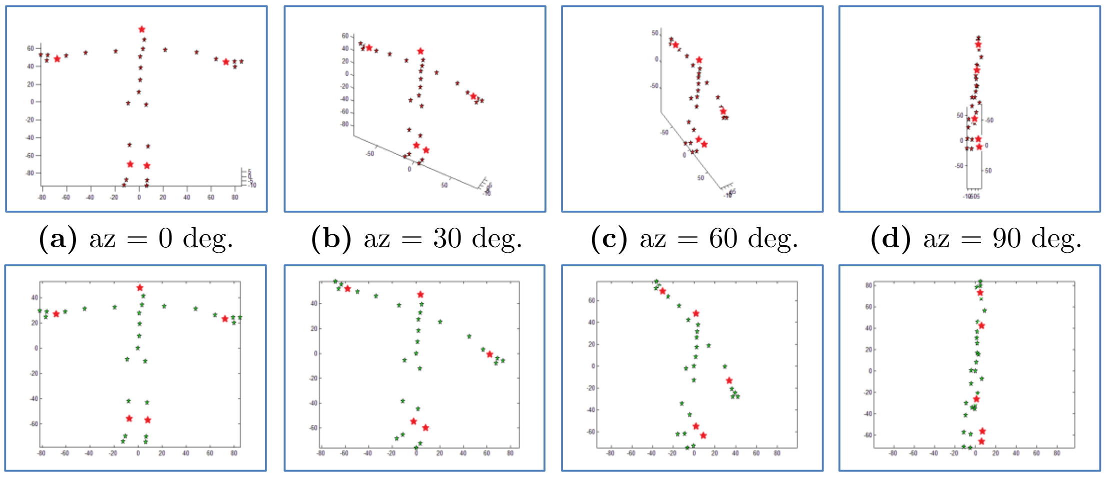


Figure 4.2: Orthographic projection camera model which we exploit for the sampling of the MoCap dataset at different view directions in order to create the 2D normalized pose space ψ.

control signal through utilizing multiple energy terms. We finally optimize the objective function by deploying the gradient decent-based optimization algorithm in order to synthesize the 3D poses frame-by-frame. We will discuss all this stuff in Section 4.4. In Section 4.5, we present quantitative and qualitative evaluation results of our proposed framework and at the end we conclude this work in Section 4.6. The overall system flow has been illustrated in Figure 4.1.

4.3 Motion Retrieval

The first step towards motion retrieval and reconstruction is the selection of those feature sets which should not only be of lower-dimensional, but can also represent the high dimensional human pose without losing significant information. We select positions of the five joints (the four end effectors and the head) for the feature sets in order to search and retrieve nearest examples from the MoCap dataset. The authors [KTWZ10] conclude in their work that the feature set based on the four end effectors and the head is the best choice especially for the real-time applications. In contrast to [KTWZ10], we are dealing with more sparse input signals that is the 2D video data rather than the 3D poses. We devise the 2D feature sets which are derived either from the synthetic examples directly or from the real video through employing SURF and MSER feature detectors and descriptors. We denote the skeleton model with **S**, the total number of all joints included in **S** with notation $\mathcal{N}_{\mathrm{j}}$, a set of joints as $\mathcal{J}$ while a set of joints involved in the feature sets are represented with notation $\mathcal{J}_{\mathcal{F}}$. In this chapter, $\mathcal{J}_{\mathcal{F}}$ consists of a set of five joints *e.g.* the four end effectors and the head. Thus, using the joint set $\mathcal{J}_{\mathcal{F}}$,



(a) az = 0 deg. **(b)** az = 30 deg. **(c)** az = 60 deg. **(d)** az = 90 deg.

(e) az = 0 deg. **(f)** az = 30 deg. **(g)** az = 60 deg. **(h)** az = 90 deg.

Figure 4.3: The representation of the 3D poses **(first row)** and the corresponding 2D poses **(second row)** obtained through orthographic projection, when the elevation angle is fixed to 45 degree and the azimuth angles are: 0 degree, 30 degree, 60 degree and 90 degree. The bigger stars with *red* color represent those body joints $\mathcal{J}_{\mathcal{F}}$ which are selected to develop the feature sets.

we develop the feature sets $\mathcal{F}_5^{\mathrm{syn}}$ which are extracted from the MoCap dataset and the feature sets $\mathcal{F}_5^{\mathrm{vid}}$ which are detected and tracked from the video input control signals.

4.3.1 Feature Set from MoCap-Data

We extract the feature sets $\mathcal{F}_5^{\mathrm{syn}}$ from the MoCap dataset in three important steps described in detail as under,

- In first step, we extract 3D positions of the four end effectors and the head from the 3D normalized poses. For a normalized pose, we consider all joint positions in the root node coordinate system. In this representation, we discard orientation and positional information in the global system—the poses might be similar independent from the actual view and place, they are captured at.
- The second step involves the projection of the 3D normalized poses onto a virtual image-plane, that is parameterized by the elevation and azimuth angles. We make use of an orthographic projection and ignore all intrinsic camera parameters. The orthographic camera model is illustrated in Figure 4.2. As a result, we obtain the 2D feature sets depending on the different view directions that are specified by the elevation and azimuth angles, the plane is parameterized with.
- Finally, in a third step, the feature set $\mathcal{F}_5^{\mathrm{syn}}$ is computed by an additional normalization step. We translate the 2D feature points to have their center of mass at the

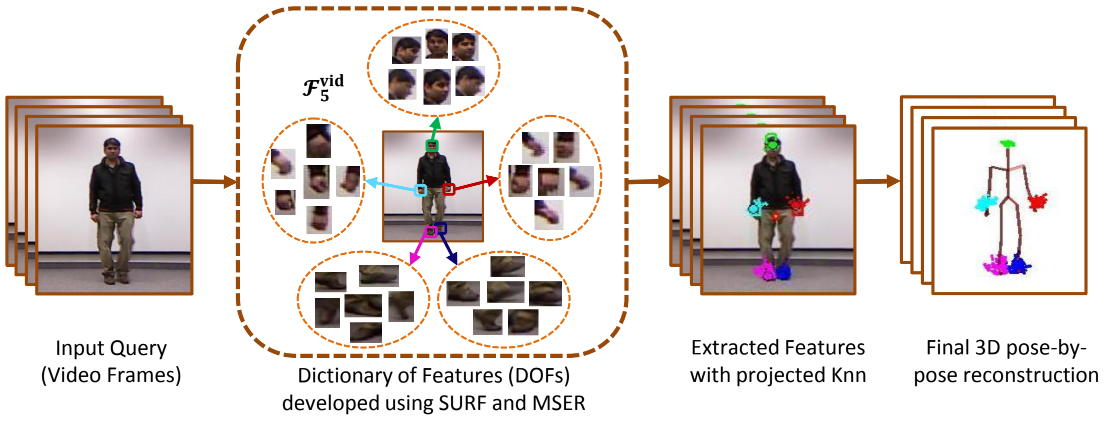


Figure 4.4: The dictionary of features (DOFs) which we have developed for the process of extracting feature sets from the video input control signals. These feature sets are then used to retrieve the nearest neighbors from the MoCap dataset, which are further exploited in the final 3D reconstruction.

origin of the coordinate system. This step is necessary to get the feature set comparable to the later described video-based feature set $\mathcal{F}_5^{\mathrm{vid}}$, where no articulated skeleton exists that could be used for normalization.

A few examples of the 3D poses with the corresponding 2D poses obtained through orthographic projection at various view directions as well as the derived feature sets are shown in Figure 4.3.

4.3.2 Feature Set from Video Data

We develop the feature sets $\mathcal{F}_5^{\mathrm{vid}}$ from the input video data, that are comparable to the feature sets $\mathcal{F}_5^{\mathrm{syn}}$ extracted from the motion capture data.

Camera Parameters. We have recorded the video sequences for input query using a Kinect RGB camera and use Kinect 3D skeleton information of the first few frames for camera calibration only. The transformation between 3D position of i^{th} joint, $X_i = [\mathsf{X}_{x,i}, \mathsf{X}_{y,i}, \mathsf{X}_{z,i}, 1]^T \in \mathbb{R}^4$, and the i^{th} joint of 2D image-based pose, $x_i = [\mathsf{x}_{x,i}, \mathsf{x}_{y,i}, 1]^T \in \mathbb{R}^3$, in homogeneous coordinate system is,

$$x_i = \mathcal{M} X_i, \tag{4.1}$$

where $\mathcal{M}$ is the *camera matrix*. The detailed version of projective Equation 4.1 can be expressed as,

$$\begin{bmatrix} \mathsf{x}_{x,i} \\ \mathsf{x}_{y,i} \\ 1 \end{bmatrix} = \mathbf{W} \left[\mathbf{R}_{(\alpha,\beta,\gamma)} \mid \mathbf{T}_{(x,y,z)} \right] \begin{bmatrix} \mathsf{X}_{x,i} \\ \mathsf{X}_{y,i} \\ \mathsf{X}_{z,i} \\ 1 \end{bmatrix}, \tag{4.2}$$

where $\left[\mathbf{R}_{(\alpha,\beta,\gamma)} \mid \mathbf{T}_{(x,y,z)} \right]$ are the *extrinsic camera parameters* which involve 3 rotational parameters (α, β and γ) and the 3 translational parameters (T_x, T_y and T_z). Under scaled orthographic projection [PMFR14, AB15], $\gamma = 0$ and $T_z = 1$. The term $\mathbf{W}$ represents the *intrinsic* or *internal* camera parameters and is illustrated as,

$$\mathbf{W} = \begin{bmatrix} s_x & \kappa & \varepsilon_x \\ 0 & s_y & \varepsilon_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

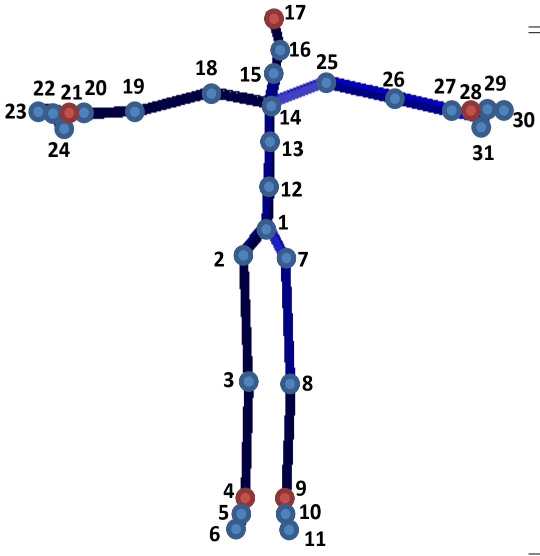
The notations f is the focal length in pixel size units, κ is the skew coefficient between x-axis and y-axis and its value is set to be zero, s_x and s_y are the scaling factors in x and y-directions, ε_x and ε_y are the principal points which are ideally considered as image centers. For square-pixels, s_x is equal to s_y. Using these values, the Equation 4.3 becomes,

$$\mathbf{W} = \begin{bmatrix} s & 0 & \varepsilon_x \\ 0 & s & \varepsilon_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \rho & 0 & \varepsilon_x \\ 0 & \rho & \varepsilon_y \\ 0 & 0 & 1 \end{bmatrix}, \quad \rho = sf. \tag{4.4}$$

As we are dealing with static camera and consider the translation zero under the assumption that the center of the mass of the 3D pose coincides the 2D pose centroid, we only have to find out ρ as unknown parameter for the intrinsic camera parameters which can be computed by the already known 2D-3D information of the first few frames.

Feature Detection and Tracking. In case of real video input, we detect and track the features of the hands, feet and the head for all video sequences[1]. For that purpose, we utilize the local feature detectors/descriptors such as SURF (Speeded Up Robust Features) [BETVG08] combined with MSER (Maximally Stable Extremal Regions) [MCUP02, DB06] and colorMSER. The detail for these local feature detectors/descriptors has been discussed in Section 2.3. We first manually annotate the positions of the hands, feet and the head in the first frame of the video and draw boxes around their boundaries. Using SURF, MSER and colorMSER feature detection techniques, the 2D

[1]We employ the four end effectors (left/right wrists, left/right ankles) and the head for the feature joints $\mathcal{J}_\mathcal{F}$ when we deal with the MoCap dataset while for the video sequences, the feature joints $\mathcal{J}_\mathcal{F}$ includes left/right hands, left/right feet and the head.



(a) Skeleton model.

Joints	Details	Joints	Details
1	Root	*17*	*Head*
2	Left Hip	18	Left Shoulder
3	Left Knee	19	Left Elbow
4	*Left Ankle*	20	Left Radius
5	Left Foot	*21*	*Left Wrist*
6	Left Toe	22	Left Hand
7	Right Hip	23	Left Fingers
8	Right Knee	24	Left Thumb
9	*Right Ankle*	25	Right Shoulder
10	Right Foot	26	Right Elbow
11	Right Toe	27	Right Radius
12	Belly	*28*	*Right Wrist*
13	Chest	29	Right Hand
14	Neck	30	Right Fingers
15	Upper Neck	31	Right Thumb
16	Lower Head		

(b) Joints' details.

Figure 4.5: **(a)** The skeleton model **S** with all $\mathcal{N}_{\mathrm{j}} = 31$ joints. The joint set $\mathcal{J}_{\mathcal{F}}$ is represented with *red* color circles. **(b)** illustrates all joints' details.

image features are extracted. These extracted features are tracked in next frames by matching them with the already extracted features of the previous frames. If the features are not matched with previously detected features, the box moves around (left, right, up and down) until the features are matched. When the features are matched, the box will update its position and move to the new position and so on. Like bags-of-words model, we develop a dictionary of features (DOFs) which maintains all the extracted features of the previous frames. The extracted features of the new frame are also added into this dictionary of features. In this way, DOFs has complete record of the features of the hands, feet and head at different positions and orientations and we can deal properly the process of matching as well as the issues that may arise due to the different orientations and positions of the hands, feet and head in different frames. An exemplary illustration of DOFs has been demonstrated in Figure 4.4.

Normalization Step. The normalization step is necessary here too for the feature sets extracted from the video data $\mathcal{F}_5^{\mathrm{vid}}$ in order to match the coordinate systems of both kinds of feature sets *i.e.* $\mathcal{F}_5^{\mathrm{vid}}$ and $\mathcal{F}_5^{\mathrm{syn}}$. We consider the center of the mass as the origin of the coordinate system by computing the mean value and translate the feature set to the origin.

4.3.3 Nearest Neighbors Search

The efficient search and retrieval of the 3D pose examples which are closest to the input control signals is the keystone in our proposed data-driven methodology. We develop an intermediate container, the *knowledge base*, through which we make search and retrieval more feasible and convenient. We define *knowledge base* as a store keeping all corresponding information needed for the efficient pose retrieval from the MoCap dataset. It is a preprocessing step that has been performed for only once.

We build up the 3D normalized pose space $\mathbf{\Psi}$ which includes the 3D feature sets extracted from the 3D normalized poses. We normalize the 3D poses by discarding orientation and translation information from the poses available in our motion capture dataset as mentioned in Section 4.3.1, so that the process of search and retrieval become free from the ambiguities that may arise due to these parameters. The 3D normalized pose space becomes the first element to be added into our intermediate container the *knowledge base*.

The input query to our reconstruction approach is either the 2D synthetic video examples or some real video sequences. For 2D input video, there exists neither any articulated skeleton for the 2D pose nor any well defined camera parameters with 11 degrees of freedom are available. Furthermore, we assume no any subject's relevant characteristics like height, weight or face directions *etc.* Having lack of these parameters, we have to search for 3D similar poses into the MoCap dataset just on the basis of a few joint locations $\mathcal{J}_{\mathcal{F}}$ of the 2D pose. To cope this, we create the 2D normalized pose space ψ by projecting the 3D normalized feature sets onto the 2D image plane using orthographic projection (see Section 4.3.1) through a number of virtual cameras, which are basically the view directions with different combinations of azimuth and elevation angles. Due to the limited memory capacity and computational resources, we create 18×7 virtual cameras by spanning the azimuth angles (0–20–340) degrees with step size 20 degree and the elevation angles (0–15–90) degrees with step size 15 degree. To this end, we have 2D normalized pose space consisting of the 2D feature sets obtained through multiple specified view directions. We also include this 2D normalized pose space into the intermediate *knowledge base*. Based on these 2D normalized pose features, we construct a kd-tree that is used for K nearest neighbor search.

Depending on the considered scenario, we extract the feature sets $\mathcal{F}_5^{\text{syn}}$ or $\mathcal{F}_5^{\text{vid}}$ from the input sequences and search into the MoCap dataset for K nearest neighbors for every single frame. We make no any assumption about the direction at which our input motion sequence was recorded during the reconstruction process. Due to the sampling of the MoCap data from different view directions, the same frame of the database might be included to the neighborhood of a query frame multiple times. This doesn't mean a disadvantage—these frames contribute stronger in the later reconstruction process. If one wants to avoid such a stronger influence on the result, the duplicates could be easily removed from the neighborhood. In our experiments, we have not found this additional step necessary. We will elaborate the influence of the size of Knn as well as the impact of sampling of the MoCap data in Section 4.5. We use ANN (Approximate Nearest Neighbor searching) C++ library [MA06] in order to search for K nearest neighbors.

The time complexity for K nearest neighbor search using kd-tree is represented as, $O(\mathsf{KM}\log(\mathsf{P}\times\mathsf{N}))$ where K is the fixed value for K nearest neighbors, M is the size (total number of frames) of the query, P is the number of 2D projections (number of virtual cameras) and N is the size (total number of frames) of the MoCap data.

4.4 Online Motion Reconstruction

The ultimate goal of our proposed reconstruction pipeline is to recover the 3D motion sequences as close to as the original motion sequences. In order to synthesize the final 3D motion sequences, we make use of the domain knowledge embedded in the motion capture dataset through *knowledge base*, which is coupled with projective constraint and the temporal coherence of the input control signals. The presented approach exploits the lazy learning where the algorithm waits for the query before the process of generalization. More precisely, all the computations in lazy learning based methods are delayed until an explicit request or a query is made. Having in hand K nearest neighbors, first a 3D local pose model is developed. Chai and Hodgins [CH05] argue in their paper that the local models are sufficient and adequate in order to develop a global model. We reconstruct the 3D motion sequences by the linear combination of the local models [CH05, BSB+07, JSMH12, WWL+14]. We formulate the process of reconstruction as energy minimization problem where the different energy terms involved in optimization ensure that the model fits even in case of some contradictory scenario like non-existence of anthropometry constraint in input 2D pose. The optimization itself is implemented using the gradient decent based non-linear method. The process of optimization for the reconstruction is the bottleneck in the performance of the system.

4.4.1 Local Model for Pose Synthesis

Safonova *et al.* [SHP04] describe in their work that the human motions can be represented efficiently in low-dimensional space. We develop the 3D local pose model in low-dimensional PCA subspace on the basis of joint angle parameterizations of K nearest neighbors which are retrieved from the MoCap data by the given 2D input feature sets at current frame t. We denote the set of joint angle parameterizations of K closest examples at current frame t as $\mathcal{Q}_t = \{\mathbf{Q}_{t,\mathsf{k}} | \mathsf{k} = 1, \ldots, \mathsf{K}\} \in \mathbb{Q}$ in quaternion pose space and set of corresponding joint positions with $\mathcal{X}_t = \{\mathbf{X}_{t,\mathsf{k}} | \mathsf{k} = 1, \ldots, \mathsf{K}\} \in \mathbb{R}$. The 3D synthesized pose, represented as $\widetilde{\mathbf{Q}}$, is the linear combination of a set of V basis vectors $\mathbf{B}_t = \{\mathbf{b}_{t,v} | v = 1, \ldots, V\}$ at frame t, which are basically the principal components computed from K nearest neighbors. The principal component coefficients are the eigenvectors with the largest eigenvalues of the covariance matrix of K nearest neighbors. The μ_t is the mean of the K nearest neighbors, $\mathcal{C}_t$ is the low-dimensional representation of the current pose in coordinates of PCA space and V is the number of basis poses. Mathematically,

$$\widetilde{\mathbf{Q}}_t = \sum_{v=1}^{V} c_{t,v} \mathbf{b}_{t,v} + \mu_t, \tag{4.5}$$

$$\widetilde{\mathbf{Q}}_t = \mathcal{C}_t \mathbf{B}_t + \mu_t. \tag{4.6}$$

In our experiments, we have found that 32 eigenposes are enough to produce very plausible results.

4.4.2 The Objective Function

We formulate the objective function with a set of four energy units to fit the 3D local linear model according to the MoCap prior in low dimensional space through optimization. These four energies are: *prior energy*, *pose energy*, *control energy*, and *smoothness energy*, which are combined together to generate the energy minimization function for motion synthesis. Mathematically,

$$\widehat{\mathbf{Q}} = \arg\min_{\widetilde{\mathbf{Q}}} (w_{pr} E_{pr} + w_{ps} E_{ps} + w_c E_c + w_s E_s), \tag{4.7}$$

where, the notations w_{pr}, w_{ps}, w_c and w_s are the weights for prior energy, pose energy, control energy and smoothness energy respectively. These weights are user defined constants and in our experiments, we found their values as: $w_{pr} = 0.5$, $w_{ps} = 1$, $w_c = 0.5$ and $w_s = 0.03$. Moreover, each energy is normalized with a normalization factor Z_t, which represents the number of elements involved in the specific energy unit at current frame t.

Prior Energy. It measures a-priori likelihood of the current synthesized pose and restricts the synthesized pose to produce acceptable results in accordance with pre-existing knowledge which is available in the MoCap dataset and is retrieved in the form of joint angle configurations $\{\mathbf{Q}_{t,\mathsf{k}} | \mathsf{k} = 1, \ldots, \mathsf{K}\}$ of K closest examples at current frame

t. Mathematically, it is computed through Mahalanobis distance as,

$$E_{pr} = \frac{1}{\sqrt{Z_t}} \cdot \|(\widetilde{\mathbf{Q}}_t - \mu_t)^T \Omega_t^{-1} (\widetilde{\mathbf{Q}}_t - \mu_t)\|^2, \tag{4.8}$$

where $\widetilde{\mathbf{Q}}_t$ is the synthesized pose, μ_t is the mean vector of K examples at frame t and $(\widetilde{\mathbf{Q}}_t - \mu_t)^T$ is the transpose of the difference between them. The term Ω_t^{-1} is the inverse of the covariance matrix of K examples which is calculated with Singular Value Decomposition (SVD).

Pose Energy. This energy unit is entertained only when the real video sequences are given as input query. We assume that in case of 2D image feature sets $\mathcal{F}_5^{\mathrm{vid}}$ given as input, we may synthesize poses which are affected by some back and forth unnecessary movements. To avoid these artifacts, we introduce the pose energy which optimize the joint positions of the synthesize pose according to the joint positions $\{\mathbf{X}_{t,\mathsf{k}} | \mathsf{k} = 1, \ldots, \mathsf{K}\}$ of the retrieved K nearest neighbors. Mathematically,

$$E_{ps} = \frac{1}{\sqrt{Z_t}} \cdot \|(\widetilde{\mathbf{X}}_t - u_t)^T \Lambda_t^{-1} (\widetilde{\mathbf{X}}_t - u_t)\|^2, \tag{4.9}$$

where $\widetilde{\mathbf{X}}_t$ is the positional representation of the synthesized pose at frame t, which we get by applying forward kinematics function on the synthesized pose $\widetilde{\mathbf{Q}}_t$. The term u_t is the mean vector and Λ_t^{-1} represents the inverse of the covariance matrix, both are computed on positional data of the K examples at frame t.

Control Energy. Control energy minimizes the distance or deviation between 2D projection of the synthesized pose $\widetilde{\mathbf{X}}_t$ and the 2D estimated feature sets of the image-based 2D pose $\mathbf{x}_t$ at current frame t. Here, we consider only those joints $\mathcal{J}_{\mathcal{F}}$ that involve for generating the feature sets *e.g.* the four end effectors and the head,

$$E_c = \frac{1}{\sqrt{Z_t}} \cdot \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} \|\widetilde{x}_{t,i} - x_{t,i}\|^2}, \tag{4.10}$$

where $x_{t,i}$ is the i^{th} joint's 2D estimated position at current frame t and $\widetilde{x}_{t,i}$ represents the i^{th} joint's position of the projected 2D pose $\widetilde{\mathbf{x}}_t$ which we compute as,

$$\widetilde{\mathbf{x}}_t = (\mathcal{I}_{\mathcal{N}_\mathsf{j} \times \mathcal{N}_\mathsf{j}} \otimes \mathcal{M}_t) \widetilde{\mathbf{X}}_t, \tag{4.11}$$

where $\mathcal{M}_t$ is the projection matrix, $\mathcal{I}$ represents the identity matrix and $\otimes$ denotes the Kronecker product.

Databases	**Details**
$\mathcal{DB}_{comp}$	It contains all motion sequences included in HDM05 MoCap dataset except the testing motions that are to be reconstructed, and is sampled with the elevation angles (0–30–90) degrees and the azimuth angles (0–30–330) degrees.
$\overline{\mathcal{DB}}_{comp}$	It includes all motion sequences of HDM05 MoCap dataset except the testing motions, while the elevation angles are set as (0–15–90) degrees and the azimuth angles are (0–20–340) degrees.
$\widehat{\mathcal{DB}}_{comp}$	It includes all motion sequences of HDM05 MoCap dataset excluding the testing motions, and the sampling is done with the elevation angles (0–15–90) degrees and the azimuth angles (0–10–350) degrees.
$\widetilde{\mathcal{DB}}_{comp}$	It contains all motion sequences of HDM05 MoCap dataset except the testing motions, with the elevation angles (0–10–90) degrees and the azimuth angles (0–10–350) degrees.

Table 4.1: The *view-based databases*: The details of different databases which we develop in terms of view directions (the virtual cameras) for quantitative evaluation.

Smoothness Energy. The smoothness energy term imposes smoothness on the synthesized pose through the temporal coherence and avoids high frequency jittering and jerkiness artifacts. We exploit previously reconstructed poses to restrict the current synthesized pose to have an impact from the already reconstructed poses. Mathematically,

$$E_s = \frac{1}{\sqrt{Z_t}} \cdot \sqrt{\|(\widetilde{\mathbf{Q}}_t - 2\widehat{\mathbf{Q}}_{t-1} + \widehat{\mathbf{Q}}_{t-2})\|^2}, \tag{4.12}$$

where $\widetilde{\mathbf{Q}}_t$ is the current synthesized pose at frames t while $\widehat{\mathbf{Q}}_{t-1}$ and $\widehat{\mathbf{Q}}_{t-2}$ are the reconstructed poses at frames $t-1$ and $t-2$ respectively.

4.5 Performance Evaluation

We evaluate the performance of our proposed approach in both ways, quantitatively as well as qualitatively. We deploy HDM05 [MRC+07] motion capture dataset in order to carry out all these experiments. This is a heterogeneous dataset which consists of 70 different motion classes performed by five different actors. It includes roughly 1500 motion clips recorded at 120Hz. We first down-sample the MoCap dataset from 120Hz to 30Hz, which results into roughly 380K 3D poses. For quantitative analysis, we design and conduct a number of experiments utilizing a variety of databases *e.g. view-based databases* and *actor-specific databases*, which are developed with respect to view directions and the performing actors respectively as described in Tables 4.1 and 4.2.

Databases	**Details**
$\overline{\mathcal{DB}}_{comp}$	It includes all motion sequences of HDM05 MoCap dataset except the testing motions that are to be reconstructed, while the elevation angles are set as (0–15–90) degrees and the azimuth angles are (0–20–340) degrees as mentioned earlier in Table 4.1.
$\overline{\mathcal{DB}}_{actorMin}$	It contains all motions of HDM05 excluding the motions of one specific actor performing in test motion sequences, *e.g.* $\overline{\mathcal{DB}}_{mmMin}$ includes all motion sequences excluding the input motions performed by the actor *mm*.
$\overline{\mathcal{DB}}_{actor}$	This database contains all motions of just one performing actor excluding the test motion sequences, *e.g.* $\overline{\mathcal{DB}}_{mm}$ includes all motion sequences performed by the actor *mm*.
$\overline{\mathcal{DB}}_{actorMirr}$	It contains only one specific actor's motions with mirroring copies as well except the test motion sequences, *e.g.* $\overline{\mathcal{DB}}_{mmMirr}$ includes all motion sequences performed by the actor *mm* plus the mirror poses are also included in the database.

Table 4.2: The *actor-specific databases*: The details of different databases which are developed in terms of performing actor for quantitative evaluation.

We build up the testing dataset which includes 2D synthetic videos of different motion classes from easy to hard motions such as: walking motions (walking 2 steps straight with left/right start, walking 4 steps straight with left/right start, walking in left/right circle *etc.*), jumping jack motions and the cartwheel motions. We generate these synthetic videos from the MoCap files using some random camera parameters. We also specified a wide range of testing view directions *i.e.* the azimuth angles = 0–5–180 degrees with 5 degree step size and the elevation angles = 0–10–90 degrees with 10 degree step size. For qualitative analysis, we record the video sequences using RGB Kinect camera. We use the skeleton model which consists of $\mathcal{N}_\mathsf{j} = 31$ joints as described in Figure 4.5.

As performance metric for quantitative evaluation, we compute *average Euclidean distance* in centimeters between joint positions of the reconstructed pose and the ground truth pose relative to the root joint position,

$$\mathcal{E} = \frac{1}{\mathsf{M}}\frac{1}{\mathcal{N}_\mathsf{j}}\sum_{t=1}^{\mathsf{M}}\sum_{i=1}^{\mathcal{N}_\mathsf{j}} \cdot \sqrt{\|(\widehat{X}_{t,i} - \widehat{X}_{t,root}) - (X_{t,i}^{gt} - X_{t,root}^{gt})\|^2}, \tag{4.13}$$

where $\widehat{X}_{t,root}$ is the root joint of the reconstructed pose $\widehat{\mathbf{X}}_t$ and the $X_{t,root}^{gt}$ represents the root joint of ground truth pose $\mathbf{X}_t^{gt}$ at current frame t.

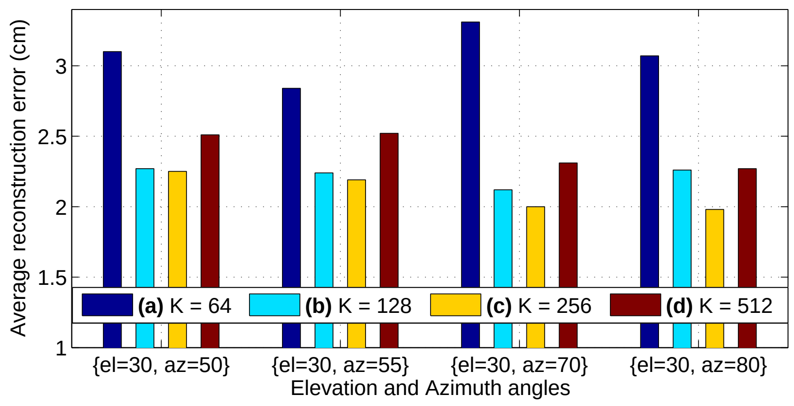


Figure 4.6: Knn Comparisons: The average reconstruction error (cm) for different numbers of nearest neighbors (K) at randomly picked four different view directions, when the walking motion sequences are given as input query.

4.5.1 Quantitative Evaluation

We evaluate our proposed framework quantitatively on synthetically generated 2D video examples. We decompose our experiments into different experimental scenarios and setups such as:

- **First**, we carry out a few pre-experiments to fix some parameters that contribute significantly in proposed methodology.
- **Second**, we evaluate our presented method for *view-based databases* which are developed through sampling of the MoCap library at various view directions (the virtual cameras). Under this setup, we not only testify the system's efficiency on different motion classes like walking, jumping jack and cartwheel motions *etc.* but also investigate that how the developed system performs at variety of test view-points.
- **Third**, we check the performance of our approach for different experimental setups and *actor-specific databases* created with respect to the performing actors.
- **Fourth**, at the end, we test effectiveness of our approach for noisy input queries with various levels of noise.

4.5.1.1 Nearest Neighbors (K)

We perform a few pre-experiments in order to set some suitable value for the parameter K. For that purpose, we check the presented reconstruction approach's efficiency for four different values of K like 64, 128, 256 and 512 at randomly picked four view directions *i.e.* {el=30, az=50}, {el=30, az=55}, {el=30, az=70} and {el=30, az=80}. We observe from the results as shown in Figure 4.6 that when the value of K is kept 256, the reconstruction error drops significantly. The value of K may vary depending on the size of the database, but in our case, the best value for K is 256 which we fix for all other experiments. We also conduct experiments with *fixed radius bound* and found that it is computationally more costly and does not produce very promising results as well. Moreover, to fix the size of the radius for all types of motion categories also create ambiguities in retrieval of the closest examples because for different motion classes the optimal radius size varies. For example, the radius size for walking motion should be smaller as compared to the radius size for jumping jack or cartwheel motions because the MoCap dataset has more 3D poses relevant to walking motion sequences. As a result, in our case, we rely on the fixed K nearest neighbor search method with the value for K equal to 256.

4.5.1.2 Database Sampling

We sample the MoCap dataset utilizing numerous virtual cameras with a variety of combinations of azimuth and elevation angles as illustrated earlier in Section 4.3.1. To check the performance of the developed algorithm with respect to the virtual cameras, we construct different *view-based databases* by exploiting a number of virtual cameras with view directions *i.e.* the azimuth angles (0–360) degrees and the elevation angles (0–90) degrees with step sizes 30, 20, 15 and 10 degrees as described in Table 4.1. We test the effectiveness of the presented system in two ways as,

- In first case, we analyse our approach with *view-based databases* for all motion classes. The results in the form of average reconstruction error for all testing viewpoints (the azimuth angles = 0–5–180, the elevation angles = 0–10–90) are reported in Table 4.3, where we discover that when we deploy more numbers of virtual cameras like in database $\widetilde{\mathcal{DB}}_{comp}$, the system performs more efficiently for all types of motion classes.
- Second, we modify the above experiment and investigate the system's behaviour at some specific testing view directions *i.e.* {el=30, az=25}, {el=30, az=35}, {el=30, az=45} and {el=30, az=55} when only jumping jack motion sequences are given as input. The results in average reconstruction error described in Figure 4.7 endorse our findings in first case, that is, with more virtual cameras ($\widetilde{\mathcal{DB}}_{comp}$), we are able to get better reconstructions with lower reconstruction error.

The database with higher numbers of virtual cameras, no doubt, produces better results but allocate more computational resources and memory space as well. So, considering both, the performance and the computational resources with memory space, we select the

Evaluation with respect to Database Sampling					
Methods	**Walk Straight**	**Walk in Circle**	**Jumping Jack**	**Cartwheel**	**Average**
$\mathcal{DB}_{comp}$	3.142	3.038	12.803	11.735	7.679
$\overline{\mathcal{DB}}_{comp}$	2.774	2.752	9.706	8.767	6.084
$\widehat{\mathcal{DB}}_{comp}$	2.573	2.466	7.992	**7.605**	5.159
$\widetilde{\mathcal{DB}}_{comp}$	**2.458**	**2.276**	**7.283**	7.666	**4.920**

Table 4.3: Comparison on *view-based databases* (see Table 4.1) for different motion classes: The average reconstruction errors in (cm) are computed at wide range of testing viewpoints (the azimuth angles = 0–5–180, the elevation angles = 0–10–90).

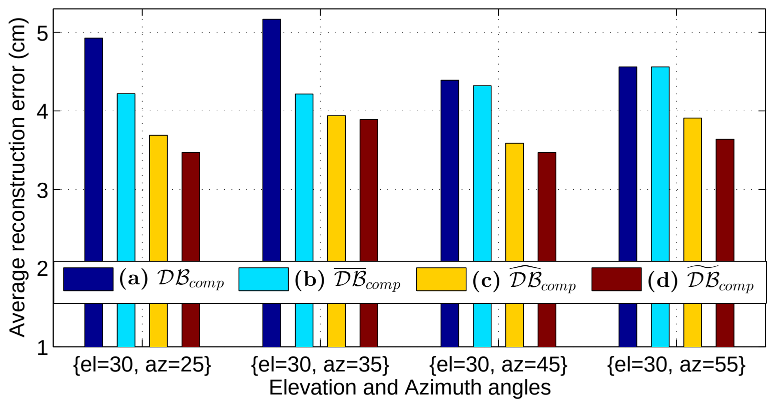


Figure 4.7: Comparison on *view-based databases* at four specified test view directions: The average reconstruction errors (cm) are computed utilizing these databases that are developed on the basis of virtual cameras as illustrated in Table 4.1, **(a)** $\mathcal{DB}_{comp}$, **(b)** $\overline{\mathcal{DB}}_{comp}$, **(c)** $\widehat{\mathcal{DB}}_{comp}$ and **(d)** $\widetilde{\mathcal{DB}}_{comp}$. For this experiment, the jumping jack motion sequences are given as input query.

database $\overline{\mathcal{DB}}_{comp}$ with the elevation angles (0–15–90) and the azimuth angles (0–20–360) to carry out further experiments.

4.5.1.3 Motion Categories

We check the presented system's performance on different types of motion categories at all test viewpoints individually utilizing diverse databases and experimental setups. The average results are reported in Table 4.3 and Table 4.4, which explore that we get very low reconstruction errors for all types of walking motion sequences while for other motion classes like jumping jack and cartwheel motions, we get comparatively higher reconstruction errors as expected. The increase in errors for these motion categories is due to the performance dissimilarities between the actors. For an example, for cartwheel motions, every actor performs it in his own way and even more, someone may not execute it accurately. That's why, it is considered as one of the toughest motions to be reconstructed. Despite of all, our approach still produces acceptable results for these motion sequences. The more reconstruction results for every test viewpoint specific to performing actors *bd* and *mm* are reported in Figure 4.8(a)–(d) and Figure 4.9(a)–(d) respectively, where an *average reconstruction error graph* for each motion class has been presented. In this graph, we represent the azimuth angles from 0 to 180 degrees with step size 5 degree along x-axis and the elevation angles from 0 to 90 degrees with step size 10 degree along y-axis. The error in the corresponding reconstruction is color coded.

4.5.1.4 View Directions

In this experimental scenario, we evaluate our approach quantitatively at a large number of test view directions (the azimuth angles = 0–5–180, the elevation angles = 0–10–90) and demonstrate that how the different view directions exert influence on overall system's performance for different kinds of motion classes like walking motions, jumping jack motions, and cartwheel motions.

- **Walking Motions.** From experiments for walking motions, it is observed that at *top view*, almost the best results are obtained in terms of reconstruction error due to the reason that in case of *top view* for walking motion, the movements of the feature joints $\mathcal{J}_{\mathcal{F}}$ are more elaborate as compared to any other view directions and as a result, the system performs well. Similarly, the *side view* also shows optimum results for walking motion sequences. In contrast, when there is a *front view*, the reconstruction error seems to be high due to the fact that at front view directions, it is difficult to capture the detailed and accurate movements of the feature joints $\mathcal{J}_{\mathcal{F}}$, especially the movements of the end effectors. In conclusion, the best suitable view for all types of walking motions is the combination of the *top views* and *side views*, when just a set of feature joints $\mathcal{J}_{\mathcal{F}}$ (the four end effectors and the head) are employed to recover the final 3D motion from the video input, otherwise the head-mounted top view motion capture is considered as the most difficult and ambiguous scenario. The worst view for the walking motions is a *front view* at lower elevation angles. All these significant conclusions are quite obvious in *average reconstruction error graph* shown in Figure 4.8(a)–(b) and Figure 4.9(a)–(b) for actors *bd* and *mm* respectively.

- **Jumping Jack Motions.** We observe just opposite behavior in jumping jack motions because the movements of the end effectors are in opposite directions to the walking motions. From the *top view* and *side view*, the high reconstruction error is executed while when there is a *front view*, the lower reconstruction error is obtained as evident from Figure 4.8(c) corresponding to the actor *bd* and Figure 4.9(c) corresponding to the actor *mm*.

Evaluation in terms of Performing Actors					
Methods	**Walk Straight**	**Walk in Circle**	**Jumping Jack**	**Cartwheel**	**Average**
$\overline{\mathcal{DB}}_{comp}$	2.774	2.752	9.706	8.767	6.084
$\overline{\mathcal{DB}}_{actorMin}$	4.033	3.722	12.989	13.811	8.638
$\overline{\mathcal{DB}}_{actor}$	2.101	2.115	5.690	7.923	4.457
$\overline{\mathcal{DB}}_{actorMirr}$	**2.043**	**2.110**	**5.299**	**7.004**	**4.114**

Table 4.4: Influence of the performing actor with *actor-specific databases* (Table 4.2). The average reconstruction error in (cm) are reported for different types of motion classes at wide range of test viewpoints (the azimuth angles = 0–5–180, the elevation angles = 0–10–90).

- **Cartwheel Motions.** For cartwheel motions, no such type of behavior like in walking motion or jumping jack motion has been observed. For all testing combinations of azimuth and elevation angles, approximately similar results in terms of average reconstruction error have been executed due to the continuously changing positions of the feature joints $\mathcal{J}_{\mathcal{F}}$. The results for cartwheel motions for the actors *bd* and *mm* can be seen in Figure 4.8(d) and Figure 4.9(d) respectively.

4.5.1.5 Performing Actors

We also analyse our proposed methodology against a variety of databases developed in terms of performing actor as explained in Table 4.2, referred as *actor-specific databases*. We deploy these databases to carry out experiments in order to investigate the impact of the concerned performing actor on the proposed system's efficiency. We conduct these experiments on synthetically generated videos and discuss one by one as under;

- $\overline{\mathcal{DB}}_{comp}$. The database $\overline{\mathcal{DB}}_{comp}$ consists of complete motion capture dataset HDM05 at sampling rate 30Hz. In this case, we only remove the testing motion sequences from the MoCap dataset, which are going to be reconstructed. We report the average reconstruction errors for all testing viewpoints (the azimuth angles = 0–5–180, the elevation angles = 0–10–90) in Table 4.4. From the results, we observe that our proposed approach performs well and produces very good results with low reconstruction errors for all types of motion categories.

  We extend the experiment to see the impact of all testing viewpoints individually under same setup. For that purpose, we input the walking sequences of the performing actor *mm* as well as the walking sequences of the performing actor *bd*. The results are presented in Figure 4.10(b) and Figure 4.11(b) respectively which explore that our proposed system executes very good results at almost all testing view angles. Similar behavior has been observed for all types of motion classes and other performing actors as well.

- $\overline{\mathcal{DB}}_{actorMin}$. We develop this database where we drop out all motion sequences of that specific performing actor whose motion sequences are given as input to the system,

  $$\overline{\mathcal{DB}}_{actorMin} = \overline{\mathcal{DB}}_{comp} - \overline{\mathcal{DB}}_{actor}. \tag{4.14}$$

  The results by taking average of all test viewpoints (the azimuth angles = 0–5–180, the elevation angles = 0–10–90) are reported in Table 4.4. From the results, we discover that the error raises a little bit up due to the skeleton discrepancies as all the motions of the input-relevant performing actor are no more the part of the database.

  We again extend the experiment to evaluate the system at every test viewpoint under this setup. We develop the database $\overline{\mathcal{DB}}_{mmMin}$ that includes all motion sequences of HDM05 excluding motion sequences of performing actor mm,

  $$\overline{\mathcal{DB}}_{mmMin} = \overline{\mathcal{DB}}_{comp} - \overline{\mathcal{DB}}_{mm}. \tag{4.15}$$

  When we query the walking sequences of the actor mm, the performance of the proposed approach drops due to the skeleton dissimilarity but the results are still acceptable as obvious in Figure 4.10(a). Similar results have been obtained for the actor bd on the database $\overline{\mathcal{DB}}_{bdMin}$ (Figure 4.11(a)),

  $$\overline{\mathcal{DB}}_{bdMin} = \overline{\mathcal{DB}}_{comp} - \overline{\mathcal{DB}}_{bd}. \tag{4.16}$$

- $\overline{\mathcal{DB}}_{actor}$. We further investigate the influence of the performing actor on system's performance by conducting a few more experiments. We develop the database $\overline{\mathcal{DB}}_{actor}$ which consists of the 3D poses of that specific performing actor who performs in the test motion. We just remove the testing input motion sequences. The reconstruction error drops a bit more in this setup as the database only includes all the motion sequences of the same actor who is present in testing motion, as evident from Table 4.4.

  To see the further impact, we testify this experimental setup at all testing elevation and azimuth angles for the actors mm and bd. We develop the databases $\overline{\mathcal{DB}}_{mm}$ and $\overline{\mathcal{DB}}_{bd}$ which consists of only the motion sequences of the actor mm and bd respectively. We just remove the testing input motion sequences only. The walking motion sequences of the actors mm and bd are given as input to the system and the results are reported in Figure 4.10(c) and Figure 4.11(c) respectively, where we have found that the reconstruction results have been improved comparatively at almost all viewpoints for both actors.

Evaluation on Noisy Inputs					
Methods	**Walk Straight**	**Walk in Circle**	**Jumping Jack**	**Cartwheel**	**Average**
$\boldsymbol{std} = 0.0$	2.774	2.752	9.706	8.767	6.084
$\boldsymbol{std} = 0.2$	2.888	2.820	9.927	8.755	6.097
$\boldsymbol{std} = 0.5$	3.028	2.930	10.237	8.915	6.277

Table 4.5: The influence of noise on the proposed system. The average reconstruction error in (cm) with different noise levels for all types of motion classes at testing viewpoints (azimuth angles = 0–5–180, elevation angles = 0–10–90).

- $\overline{\mathcal{DB}}_{actorMirr}$. For completeness, we also carry out experiments with the database $\overline{\mathcal{DB}}_{actorMirr}$ which is basically the database $\overline{\mathcal{DB}}_{actor}$ combined with the mirrored 3D poses. The error drops a little bit more as the database contains more 3D poses with mirrored copies.

  To see the influence at all testing viewpoints, we build up the database $\overline{\mathcal{DB}}_{mmMirr}$ where the database includes the motions of the actor *mm* and the mirrored 3D poses. In this scenario, the error decreases a bit more and we acquire the best reconstruction result with the same input of walking motions of the actor *mm* as obvious from Figure 4.10(d). Similar results are found for the actor *bd* on the database $\overline{\mathcal{DB}}_{bdMirr}$ (see Figure 4.11(d)).

4.5.1.6 Noisy Inputs

The predictions of the joint positions of the 2D pose in a real video are often noisy. In this regard, we also test our proposed approach on noisy input queries. We add different levels of white Gaussian noise with standard deviation ($\boldsymbol{std} = 0.0$, $\boldsymbol{std} = 0.2$, $\boldsymbol{std} = 0.5$) and check the system's performance. The noise with standard deviation ($\boldsymbol{std} = 0.0$) means no any noise added to the input data. The results are reported in Table 4.5, where it is quite obvious that our proposed methodology produces very good results even in case of noisy inputs.

4.5.2 Qualitative Evaluation

For qualitative evaluation, we testify the performance of our proposed algorithm on a number of monocular video sequences. For that purpose, we have recorded our own video sequences for different motion classes like jumping jack motions, grabbing motions and jogging motions *etc.* using Kinect RGB camera, which records videos at frame rate 30 fps with resolution 587×440.

We deploy the database $\overline{\mathcal{DB}}_{comp}$, when the real video sequences are given as input. There is no any synchronization between the 3D MoCap dataset with the recorded video sequences. Furthermore, the performing actors in the videos are different from the actors performed in the MoCap dataset. The video input sequences are first pre-processed in order to get the relevant feature sets required for the input query (Section 4.3.2). A few exemplary qualitative results of the reconstructions are presented in Figure 4.12. Overall we get very good results for almost all video sequences, even in case of outliers detected due to some noisy data. The noisy data is because of the uncertainty and ambiguities that may arise during detection or tracking of the 2D video-based feature sets $\mathcal{F}_5^{\mathrm{vid}}$. For example in a few frames, the features cannot be detected at all as a result of self-occlusion, double counting, illumination, blurring effects or the continuous variations in positions and orientations of the hands or feet. Moreover, sometime movements of the hands and feet are inconsistent *e.g.* the hands or feet move very fast in a frame as compared to the movement in previous frames which leads towards mistracking. All these factors may create inaccurate and erroneous feature sets which causes higher 3D reconstruction errors. As a solution, we annotate a few specific key frames manually in order to acquire somehow more accurate 2D video-based feature sets from the real input videos.

4.6 Conclusion

In this chapter, we have presented an efficient model-based approach to retrieve and reconstruct human motions from different types of 2D input control signals. For search and retrieval of similar examples from the MoCap dataset, we introduce *knowledge base* which consists of the 3D normalized pose space and the corresponding 2D normalized pose space. The presented approach exploits multiple energy terms to reconstruct full body human motions efficiently in a real time even when a low-dimensional 2D feature sets are given as input query which is either extracted from the 2D synthetic data or the real monocular video sequences. We have testified the robustness of our methodology on several databases and experimental setups designed with respect to performing actors, viewpoints and the noisy input using various types of human motion classes *i.e.* walking, cartwheel, jumping jack, jogging and grabbing motions. Our proposed system is fast enough and performs reconstruction approximately 5–8 frames per second.

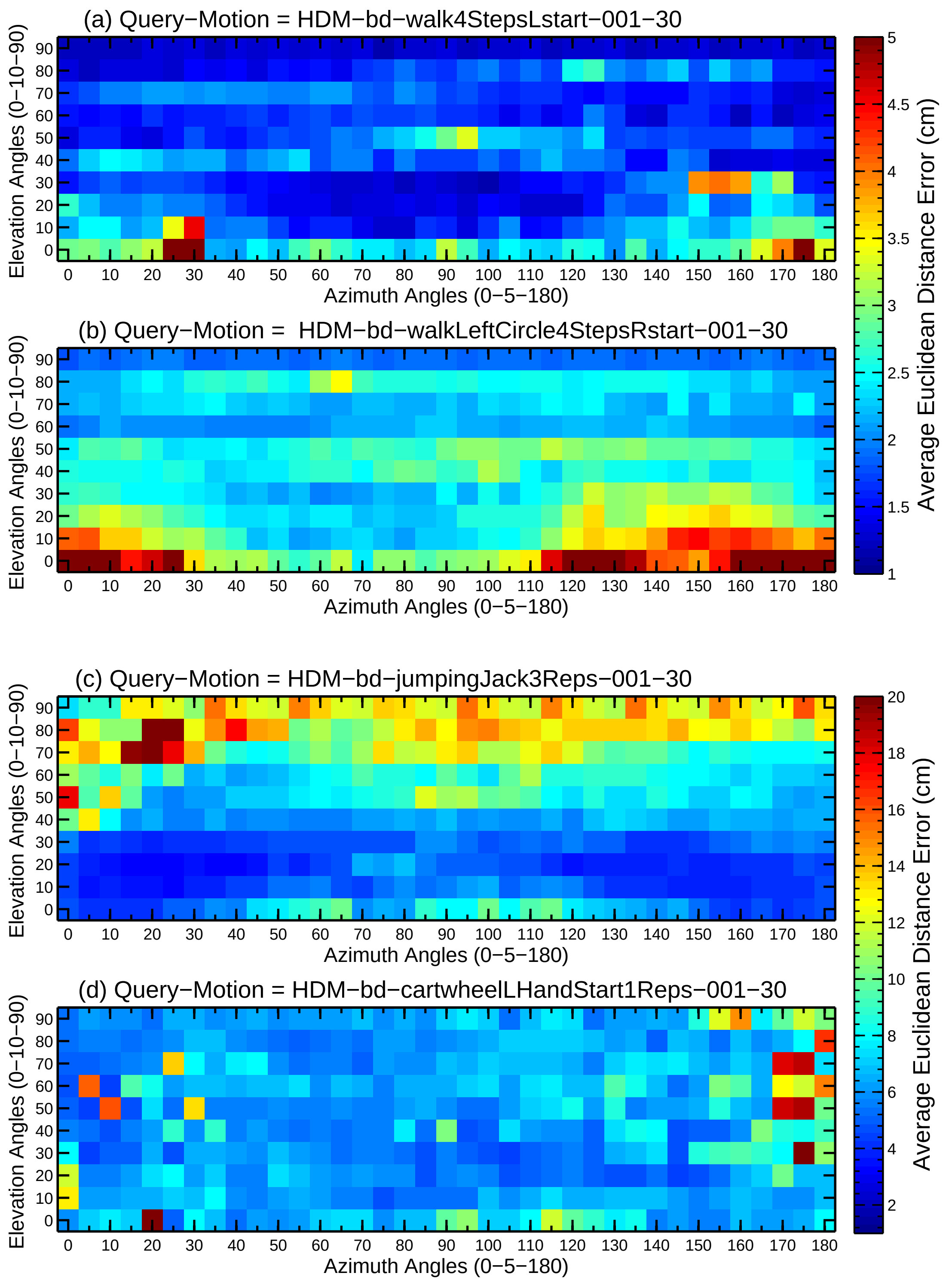


Figure 4.8: Comparison on different motion classes for the actor *bd*. The average reconstruction errors (cm) at test view directions (the azimuth angles = 0–5–180, the elevation angles = 0–10–90), when different types of motions are given as input query to the proposed system *e.g.*: **(a)** walking in straight motion; **(b)** walking in circle motion; **(c)** jumping jack motion; **(d)** cartwheel motion.

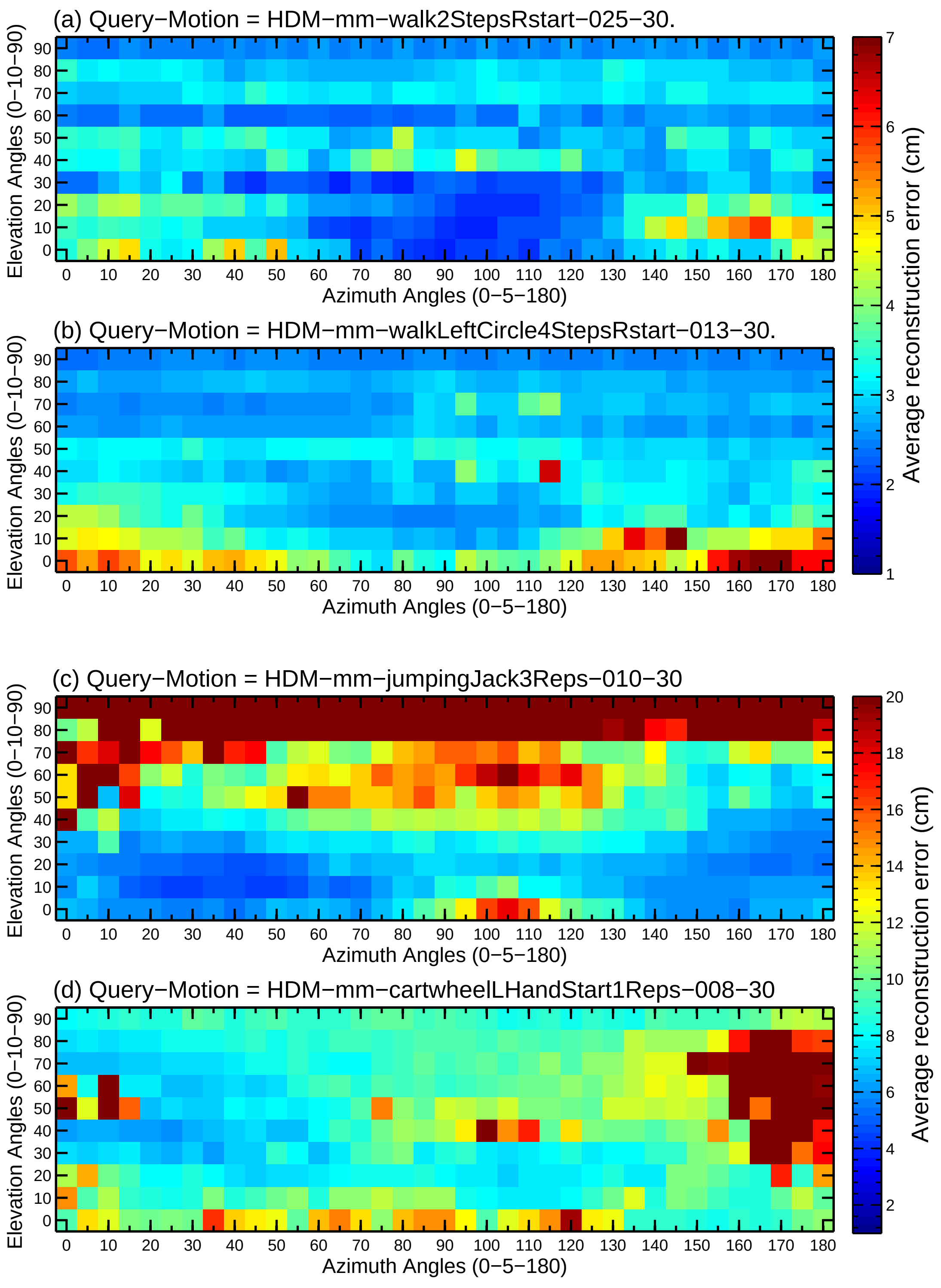


Figure 4.9: Comparison on different motion classes for the actor *mm*. The average reconstruction errors (cm) at test view directions (the azimuth angles = 0–5–180, the elevation angles = 0–10–90), when different types of motions are given as input query to the proposed system *e.g.*: **(a)** walking in straight motion; **(b)** walking in circle motion; **(c)** jumping jack motion; **(d)** cartwheel motion.

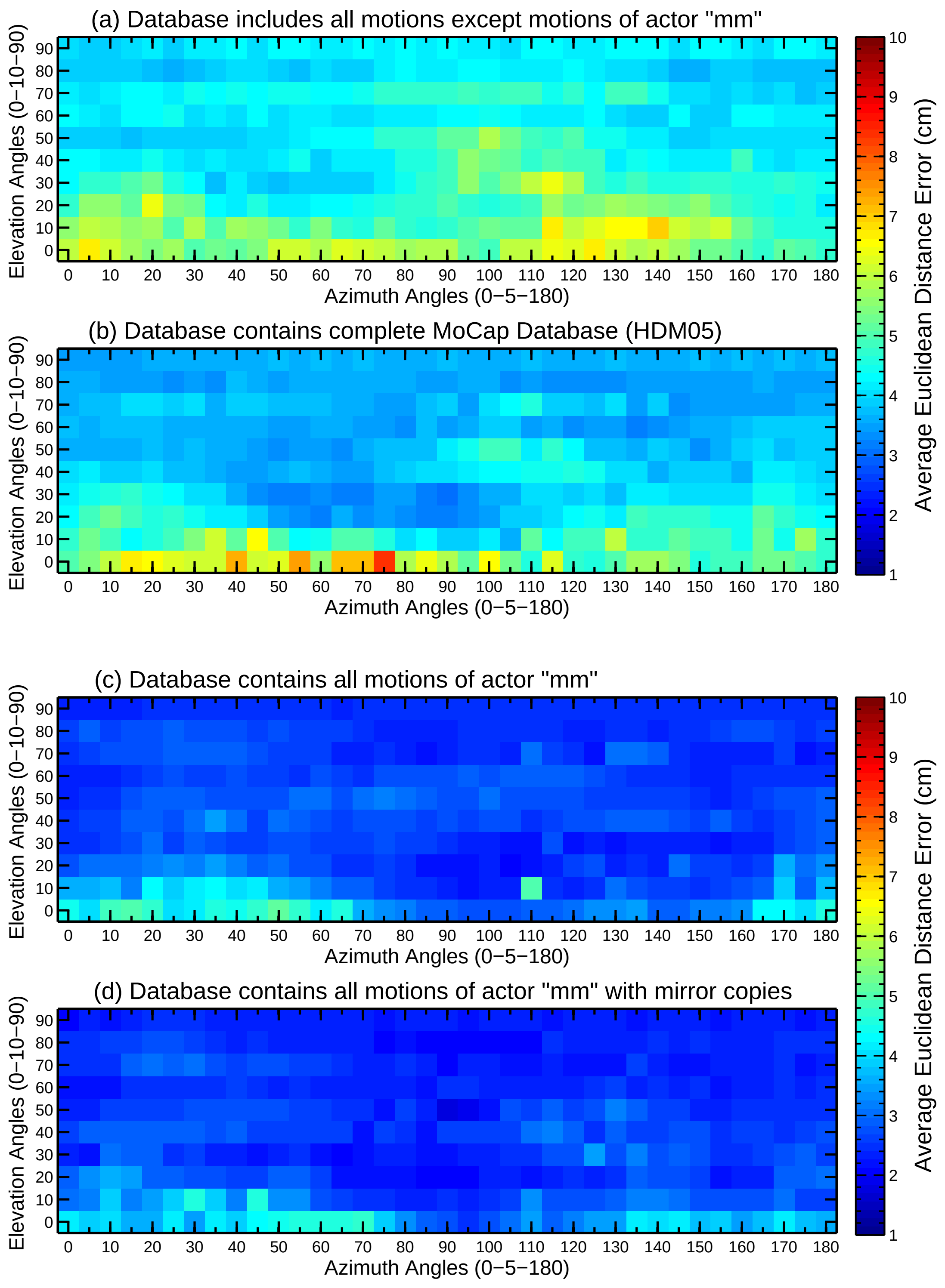


Figure 4.10: Influence of the performing actor *mm* with *actor-specific databases* (Table 4.2). The average reconstruction errors (cm) at different view directions, when the walking motion sequences performed by actor *mm* are given as input query and the employed databases are: **(a)** $\overline{\mathcal{DB}}_{mmMin}$; **(b)** $\overline{\mathcal{DB}}_{comp}$; **(c)** $\overline{\mathcal{DB}}_{mm}$; **(d)** $\overline{\mathcal{DB}}_{mmMirr}$.

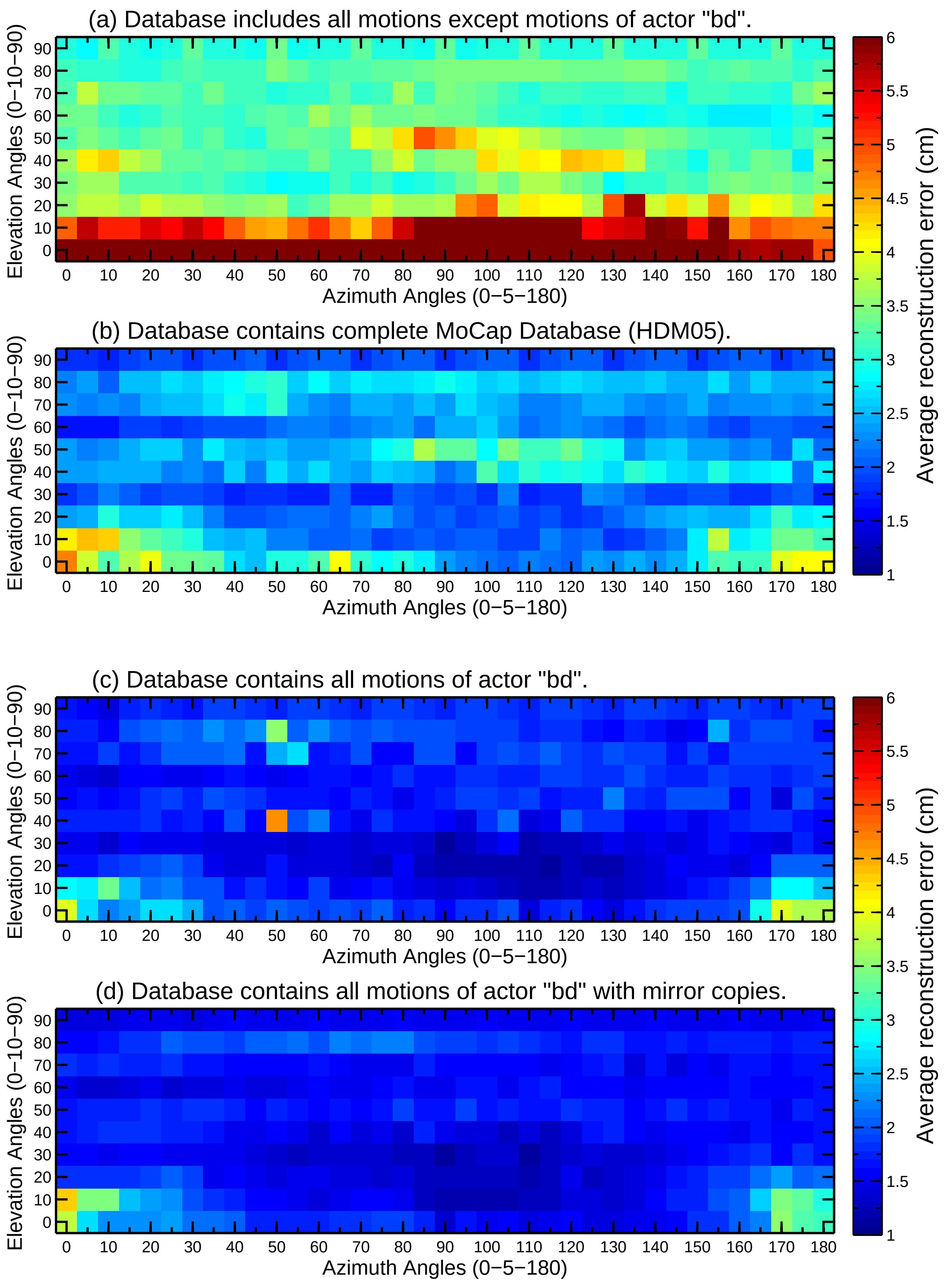
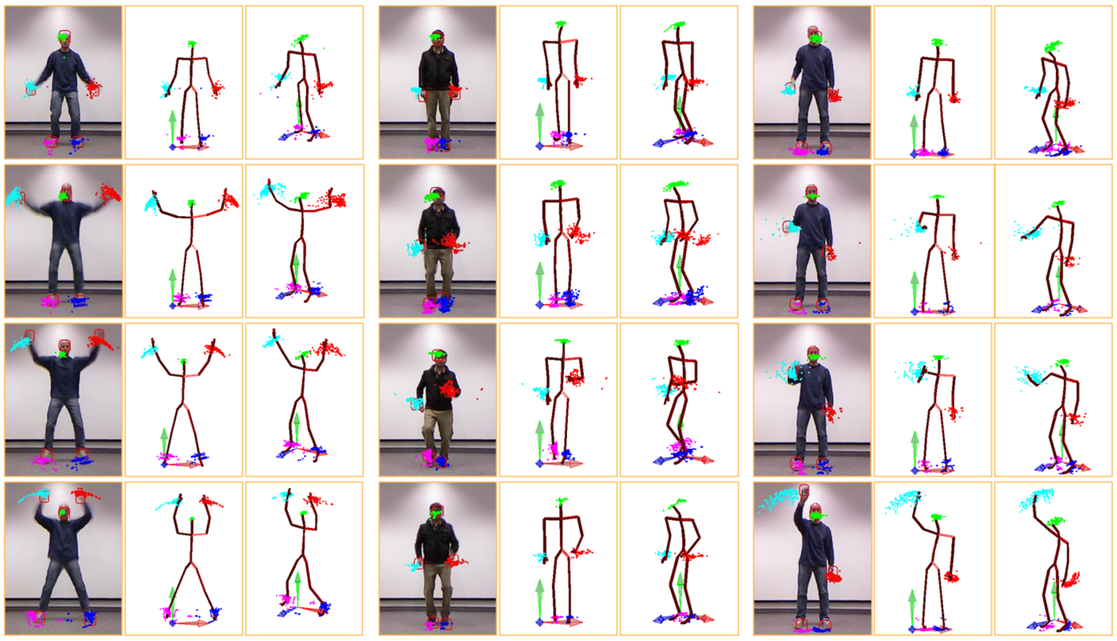


Figure 4.11: Influence of the performing actor *bd* with *actor-specific databases* (Table 4.2). The average reconstruction errors (cm) at different view directions, when the walking motion sequences performed by actor *bd* are given as input query and the employed databases are: **(a)** $\overline{\mathcal{DB}}_{bdMin}$; **(b)** $\overline{\mathcal{DB}}_{comp}$; **(c)** $\overline{\mathcal{DB}}_{bd}$; **(d)** $\overline{\mathcal{DB}}_{bdMirr}$.



(a) Jumping Jack motion **(b)** Jogging on Spot motion **(c)** Grabbing Top motion

Figure 4.12: Qualitative reconstruction results of different types of motion classes with the extracted K nearest neighbors from the motion capture dataset in case of real video input query. **(first column)** shows video frames with the extracted 2D feature sets and the projected K nearest neighbors; **(second column)** represents the relevant 3D reconstructed motion frames with the retrieved K nearest neighbors; **(third column)** corresponds to the 3D reconstructed motion frames at other viewpoints.

5 Motion Tracking and 3D Reconstruction

An experiment is a question which science poses to Nature, and a measurement is the recording of Nature's answer.

Max Planck

The chapter at hands presents a novel data-driven framework for 3D full body human motion reconstruction from a static monocular video data, which is basically the enhancement of the previously proposed approach presented in Chapter 4.

5.1 Introduction

Recall that in previous chapter, we have proposed a novel data-driven framework for 3D full body human motion reconstruction from the video data, which we improve in this chapter in many significant ways: (i) We make search and retrieval strategy more robust and efficient using the temporal coherence of the input control signals through a graphical structure *online lazy neighbourhood graph* (OLNG) [TZK+11]. We adapt this graphical structure to work with the domain of video-based control input signals, which is a more sparse, complex and challenging possible scenario. (ii) We utilize the low-level image based feature detectors and descriptors *e.g.* SURF, MSER and colorMSER for the process of 2D feature detection and tracking, which is further stabilized through the high-level 3D prior knowledge obtained from the MoCap dataset in the form of K examples closest to the control input video signals. In this way, we can handle occlusion, illumination and blurring artifacts in a more efficient way. (iii) We update the camera parameters frame-by-frame using the synthesized poses, the sampling of MoCap dataset and the temporal information. (iv) We learn a low-dimensional local 3D pose model from the optimal and weighted nearest neighbors $\widehat{\mathsf{K}}_w$. We pick up the best closest examples from K nearest neighbors and weight these optimal nearest neighbors according to the costs associated with the edges in OLNG. (v) At the end, we improve our reconstruction methodology by applying the kernel based approach [TZK+11] in order to estimate the probability density of the MoCap priors.

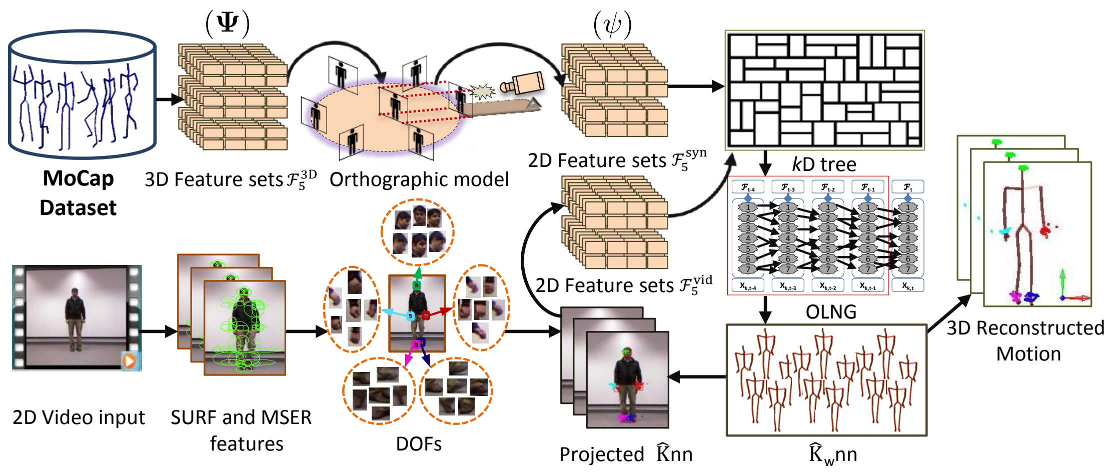

(Ψ)
(ψ)
MoCap Dataset
3D Feature sets $\mathcal{F}_5^{3D}$
Orthographic model
2D Feature sets $\mathcal{F}_5^{syn}$
kD tree
2D Feature sets $\mathcal{F}_5^{vid}$
OLNG
3D Reconstructed Motion
2D Video input
SURF and MSER features
DOFs
Projected $\widehat{\mathsf{K}}$nn
$\widehat{\mathsf{K}}_w$nn

Figure 5.1: An overview of the proposed system's pipeline with the main components involved in the system: We develop the 3D normalized pose space $\boldsymbol{\Psi}$ and then sample this $\boldsymbol{\Psi}$ at different view directions by orthographic projection to generate the 2D normalized pose space ψ. From ψ, the kd-tree data structure is built. The 2D input video is given to the system, where the features are detected and tracked using SURF and MSER feature detectors by developing a dictionary of features (DOFs) to retrieve Knn from the MoCap data. We here develop *online lazy neighbourhood graph* (OLNG) to pick up the most optimal $\widehat{\mathsf{K}}$nn. These $\widehat{\mathsf{K}}$nn are projected onto image plane to make detection and tracking more efficient and robust. We weight $\widehat{\mathsf{K}}$nn according to the costs associated with the edges in OLNG and referred as $\widehat{\mathsf{K}}_w$nn. At the end, the reconstruction is performed by exploiting these optimal and weighted $\widehat{\mathsf{K}}_w$nn.

Similar to previously presented approach in Chapter 4, we develop a *knowledge base* which includes the 3D normalized pose space $\boldsymbol{\Psi}$ and the synchronized 2D normalized pose space ψ which is generated by sampling of the 3D normalized poses at different view directions (for detail see Section 4.3). After extracting the suitable feature sets from both, the input control signals and the motion capture dataset, we perform efficient similarity search and retrieve nearest neighbors from the MoCap data. For that purpose, we construct data structures like kd-tree and *online lazy neighbourhood graph*. Finally, the 3D motion sequences are reconstructed by non-linear energy minimization that takes into account multiple prior terms. We evaluate our developed algorithm on the real video input data as well as on synthetically generated 2D input video signals. The overall system overview is illustrated in Figure 5.1.

We organize the chapter as: in Section 5.2, we focus on how we detect and track the features from the video sequences. Section 5.3 explores the details about the estimation of camera parameters. The complete details about the motion retrieval and OLNG are elaborated in Section 5.4, while Section 5.5 illustrates the proposed reconstruction methodology. The evaluation of the presented approach and the conclusion are discussed in Section 5.6 and Section 5.7 respectively.

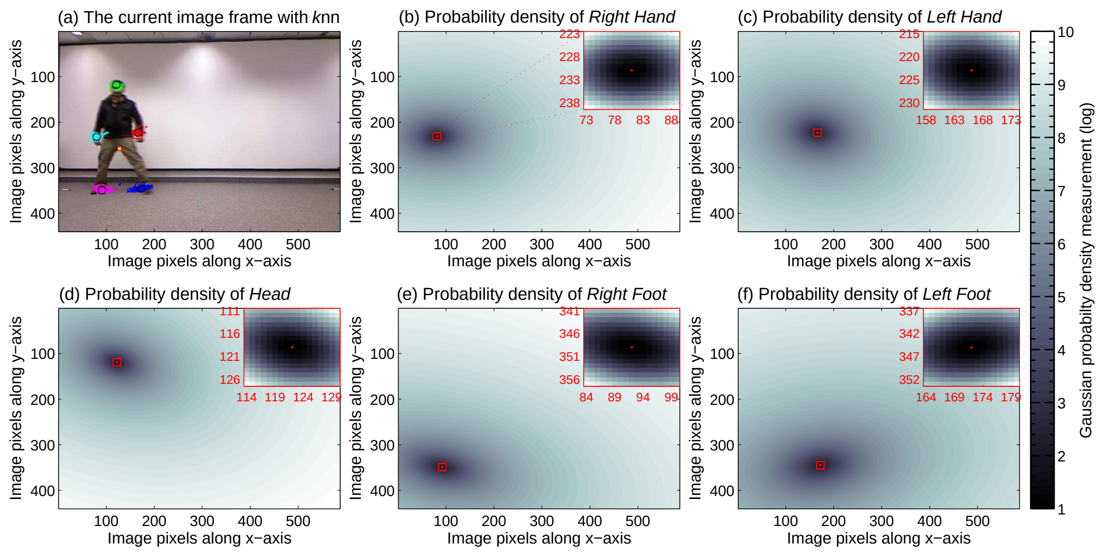
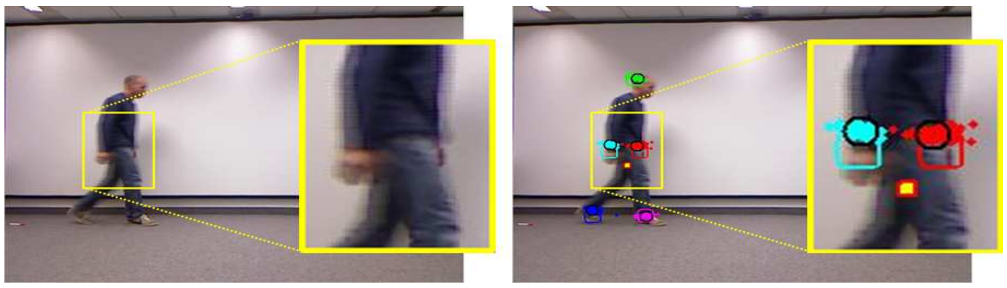


Figure 5.2: The probability densities for the hands, feet and the head used in the process of feature detection and tracking. The current input video frame with the projected $\widehat{\mathsf{K}}$nn is shown in **(a)**, while the probability density measurements for the hands, feet and the head are presented in: **(b)** *right hand*. **(c)** *left hand*. **(d)** *head*. **(e)** *right foot*. **(f)** *left foot*.

5.2 Feature Design and Tracking

In this section, we elaborate that how we design and extract the feature sets and exploit them to search into the motion capture library for the motion segments that are very closest to the input video sequences. From MoCap dataset, we extract the 2D feature set $\mathcal{F}_5^{\text{syn}}$ using the four end effectors and the head (see Section 4.3.1). Similarly, in case of real video query, we detect and track the positions of the hands, feet and the head in order to develop the feature sets $\mathcal{F}_5^{\text{vid}}$. In order to accomplish this, we employ the low-level image based feature detectors/descriptors *e.g.* SURF, MSER and colorMSER, leveraged with the high-level 3D prior knowledge exists in the MoCap dataset. With 3D MoCap priors, we refine the 2D estimated joint positions in order to develop more stable and accurate 2D feature sets $\mathcal{F}_5^{\text{vid}}$. First, we extract features by utilizing these low-level detectors/descriptors and develop a dictionary of features (DOFs) as illustrated in Section 4.3.2, which are further refined and stabilized by $\widehat{\mathsf{K}}$ nearest examples. We retrieve Knn from the MoCap data through the *knowledge base* and pick up the most optimal $\widehat{\mathsf{K}}$nn utilizing OLNG which we will discuss in Section 5.4. To this end, we have the retrieved optimal $\widehat{\mathsf{K}}$ nearest neighbors which are projected onto the current image frame by estimated camera parameters (Section 5.3). We formulate Bayes decision function D_t to obtain similar 2D feature patterns $\widetilde{\mathbf{x}}$ from the current image frame t of the input video,

$$D_t(\widetilde{\mathbf{x}}_t) = P(\widetilde{\mathbf{x}}_t|\mathbf{x}_{t1}, \ldots, \mathbf{x}_{t\widehat{\mathsf{K}}})P(\mathbf{x}_{t1}, \ldots, \mathbf{x}_{t\widehat{\mathsf{K}}}), \tag{5.1}$$



(a) Image with occlusion of left hand. **(b)** Occlusion handling with $\widehat{\mathsf{K}}$nn.

Figure 5.3: The process of occlusion handling by using the 3D prior knowledge embedded in MoCap dataset. **(a)** The image frame with occlusion of left hand which is totally disappeared. **(b)** The image frame illustrates the tackling of the occlusion by projection of the 3D $\widehat{\mathsf{K}}$nn, retrieved from the MoCap dataset.

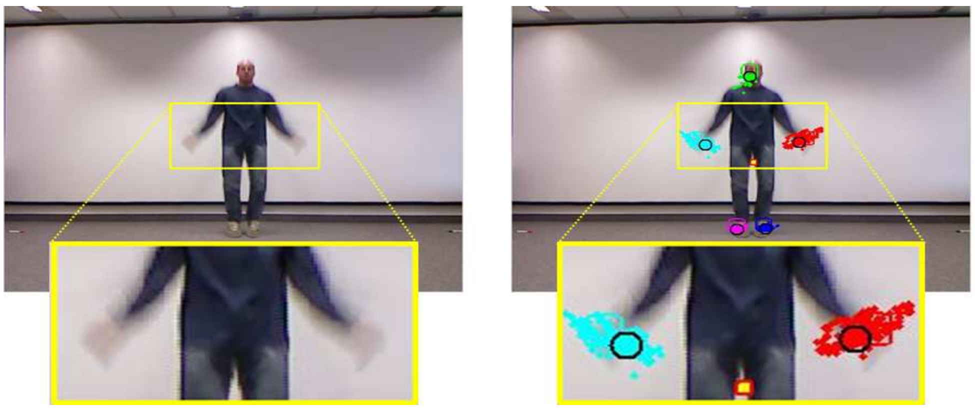
where $P(\mathbf{x}_{t1}, \ldots, \mathbf{x}_{t\widehat{\mathsf{K}}})$ is the prior probability of the projected $\widehat{\mathsf{K}}$ closest examples which we consider equally likely and $P(\widetilde{\mathbf{x}}_t | \mathbf{x}_{t1}, \ldots, \mathbf{x}_{t\widehat{\mathsf{K}}})$ is the h-dimensional Gaussian probability density function of the pattern vector and is calculated with Mahalanobis distance as,

$$P(\widetilde{\mathbf{x}}_t | \mathbf{x}_{t1}, \ldots, \mathbf{x}_{t\widehat{\mathsf{K}}}) = \frac{1}{(2\pi)^{\frac{h}{2}} \sqrt{|\Omega_t|}} \cdot e^{-\frac{1}{2}(\widetilde{\mathbf{x}}_t - \mu_t)^T \Omega_t^{-1} (\widetilde{\mathbf{x}}_t - \mu_t)}, \tag{5.2}$$

where $|\Omega_t|$ is the determinant of the covariance matrix Ω_t, μ_t is the mean vector at current frame t, $(\widetilde{\mathbf{x}} - \mu_t)^T$ is the transpose of the difference between the features' pattern vector $\widetilde{\mathbf{x}}$ and the mean vector μ, and h is the dimensions of the feature vector $\widetilde{\mathbf{x}}$. A few examples of the Gaussian probability density for each end effector (the right/left hands and the right/left feet) and the head have been shown in Figure 5.2, where the probability density in log has been color coded with darker region corresponds to higher probability density. On the basis of Bayes decision function, we select those pixels of the image frame for the features that execute the largest Bayes decision value. In the end, we combine the features detected through low-level feature detection techniques and the features estimated through projection of high-level MoCap priors, and weight them as,

$$\mathcal{F}_5^{\text{vid}} = w_1 \Theta + w_2 \Upsilon, \tag{5.3}$$

where Θ represents the features extracted through SURF, MSER and colorMSER, Υ represents the features which we get from projection of $\widehat{\mathsf{K}}$ closest examples while w_1 and w_2 are the user defined weights and their values are fixed 0.6 and 0.4 respectively in our experiments. The continuous process of detection and tracking is performed by detecting the feature sets of the current image frame, by matching them with the already extracted feature sets of the previous frames collected in DOFs, by projecting the 3D $\widehat{\mathsf{K}}$ closest examples and by estimating the highest probability of the 2D joint positions derived from $\widehat{\mathsf{K}}$nn projection. We combine all these significant steps together to extract more



(a) Image with blurring effects. **(b)** Blur handling with $\widehat{\mathsf{K}}$nn.

Figure 5.4: The process of handling blurring effect by using the 3D MoCap priors. **(a)** The image frame showing blurring effect for the both hands. **(b)** The image frame elaborates the handling of blurring effect through the projection of the 3D $\widehat{\mathsf{K}}$ closest examples.

stable and accurate feature sets $\mathcal{F}_5^{\text{vid}}$. A few examples related to the benefits obtained through MoCap priors in the process of detection and tracking have been elaborated in Figure 5.3 and Figure 5.4. The details will be discussed in Section 5.6.2.

5.3 Camera Model

We estimate camera parameters through the estimated 2D feature sets and the *knowledge base* which contains the 3D as well as 2D normalized pose spaces, represented as($\boldsymbol{\Psi}$) and (ψ) respectively. The camera projection matrix $\mathcal{M}$ consists of intrinsic parameters $\mathbf{W}$ and the extrinsic parameters with rotation $\mathbf{R}_{(\alpha,\beta,\gamma)}$ and the translation $\mathbf{T}_{(x,y,z)}$,

$$\mathcal{M} = \mathbf{W}\left[\mathbf{R}_{(\alpha,\beta,\gamma)} \mid \mathbf{T}_{(x,y,z)}\right]. \tag{5.4}$$

For intrinsic camera parameters, we have computed the focal length by employing the 3D (Kinect 3D skeleton) and 2D information of first few frames. The skew coefficient is fixed to be zero. The scaling factor s is updated across the M number of video query frames. We consider the mean of feature sets $\mathcal{F}_5^{\text{vid}}$ as the principal points $(\varepsilon_x, \varepsilon_y)$ and assume that the center of the mass of 3D pose corresponds to 2D pose centroid, which are updated regularly across the M number of video query frames. In case of extrinsic camera parameters, we estimate translation information $\mathbf{T}_{(x,y,z)} = \{T_{x,1}, \ldots, T_{x,\mathsf{M}}, T_{y,1}, \ldots, T_{y,\mathsf{M}}, T_{z,1}, \ldots, T_{z,\mathsf{M}}\}$ through regularly updated principle points, scaling factors and the focal length. Under scaled orthographic projection [PMFR14, AB15], $s_x = s_y$, T_z becomes equal to 1 and $\gamma = 0$.

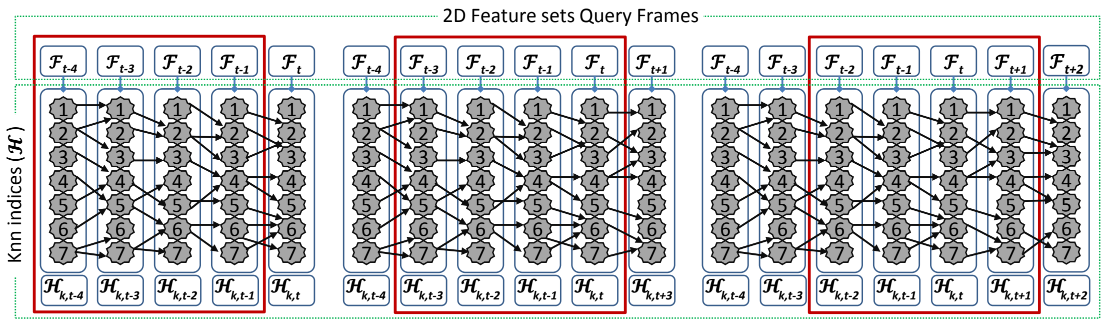


Figure 5.5: An exemplary illustration of online lazy neighbourhood graph developed on the basis of $\mathsf{K} = 7$ nearest neighbor indices $\mathcal{H}$ retrieved using 2D feature sets ($\mathcal{F}_5^{\text{vid}}$ or $\mathcal{F}_5^{\text{syn}}$) query inputs. The window with size $\mathcal{Z} = 4$ is represented with *red* box which moves on all query frames.

We estimate orientation through sampling of MoCap dataset by a simple voting strategy. For that purpose, we exploit *knowledge base* which contains information about different view directions (the azimuth and the elevation angles) in the 2D normalized pose space (ψ). Having in hands the selected optimal $\widehat{\mathsf{K}}$nn (Section 5.3), we build a histogram on the basis of the indices $\mathcal{H}$ of $\widehat{\mathsf{K}}$nn. The top three peaks of the histogram are selected to be the candidates of the current azimuth and elevation angles of the performing actor. We have found from the experiments that mostly the simple majority vote yields acceptable results. We further constrain the selection by the temporal coherence of the already selected angles. For an instance, the azimuth angle which is very close to the already selected azimuth angles for previously two frames, has higher probability to be picked up among the other candidates. Finally, we apply Gaussian low-pass filter in order to smoothen these selected angles.

5.4 Motion Retrieval

We develop an intermediate container, the *knowledge base* as described in Section 4.3.3, which contains the 3D normalized pose space $\mathbf{\Psi}$ as well as the corresponding 2D normalized pose space ψ developed through 36×7 numbers of virtual cameras with the azimuth angles (0–10–350) degrees and the elevation angles (0–15–90) degrees. In order to retrieve similar poses from the MoCap database, the 2D feature sets extracted either from the real video data or from the synthetically generated video sequences, are given as input to the developed system. For similarity search into the MoCap database, we develop a kd-tree data structure which is built upon the 2D normalized pose space ψ, and a graph structure, *online lazy neighbourhood graph* [TZK+11]. In contrast to [TZK+11], we are working on a sparse continuous streaming of 2D input video data rather than using an accelerometer data. In our case, the graph is built up on the indices $\mathcal{H}$ of

Knn retrieved through the input 2D feature sets and we do not allow the skipping of frame in our specified step sizes. The OLNG graph is the acyclic directed graph which utilizes the temporal coherence of the input video control signals. The online version of lazy neighbourhood graph imposes the fact that there is no any need to construct whole graph structure for every frame cycle, rather it is more efficient to build the graph incrementally considering previously constructed paths of the minimum cost [TZK+11]. In this context, the OLNG is developed incrementally with window size $\mathcal{Z} = 8$. To this end, we have the retrieved K closest examples with indices $\mathcal{H}$ on which basis we develop the OLNG with an array of size $(\mathcal{Z} \times \mathsf{K})$, where each retrieved example is considered as a node. We generate the weighted directed edges between these nodes, which ensure the monotonicity and create the valid continuations remaining within the size of the array. We allow the step size by exploiting the concept of dynamic time warping (DTW), where a step may be a horizontal (0,1), a diagonal (1,1), or a vertical step (1,0). Each edge is associated with a cost by computing the distance between nodes in terms of similarities between the feature sets. In construction of OLNG, we consider only those paths which have minimum costs. An exemplary illustration of OLNG is presented in Figure 5.5.

From the retrieved K nearest neighbors, we select the best and optimal closest examples, represented as $\widehat{\mathsf{K}}$, by considering the step sizes and the minimum costs associated with the paths in OLNG. Furthermore, we weight each pose in the selected best $\widehat{\mathsf{K}}$ closest examples, which we compute on the basis of the associated costs and normalize them as,

$$w_{t,\mathsf{k}} = 1 - \frac{G_{t,\mathsf{k}} - \min(G_t)}{\max(G_{t,\mathsf{k}} - \min(G_t))} \qquad \mathsf{k} = 1, 2, \dots, \widehat{\mathsf{K}}. \tag{5.5}$$

where $w_{t,\mathsf{k}}$ represents the weights and the term $G_{t,\mathsf{k}}$ denotes the associated costs for the selected paths at current frame t. In our experiments, the K nearest neighbors is fixed to be 2^{12} and from which we select just only 256 $\widehat{\mathsf{K}}$ best examples.

5.5 Online Motion Reconstruction

We leverage the motion capture data as a source of prior knowledge to resolve the depth ambiguities and finally infer the high-dimensional human motions. To this end, we retrieve similar poses from the MoCap database which is in the form of joint angle configurations $\mathcal{Q}_t = \{\mathbf{Q}_{t,\mathsf{k}} | \mathsf{k} = 1, \dots, \mathsf{K}\}$ with corresponding positions $\mathcal{X}_t = \{\mathbf{X}_{t,\mathsf{k}} | \mathsf{k} = 1, \dots, \mathsf{K}\}$ at current frame t. A local 3D pose model is built from the optimal weighted nearest neighbors, $\widehat{\mathsf{K}}_w$. The 3D pose $\widetilde{\mathbf{Q}}$ is synthesized by a linear combination of set of V basis vectors $\mathbf{B}_{w,t} = \{\mathbf{b}_{w,t,v} | v = 1, \dots, V\}$ at frame t, which are the principal components computed from the $\widehat{\mathsf{K}}_w$ nearest neighbors. The μ_w is the mean of the weighted $\widehat{\mathsf{K}}_w$ nearest neighbors, $\mathcal{C}$ is the low-dimensional current pose in the coordinates of PCA space. Mathematically,

$$\widetilde{\mathbf{Q}}_t = \sum_{\mathsf{k}=1}^{\widehat{\mathsf{K}}_w} w_{\mathsf{k}} \cdot \sum_{v=1}^{V} c_{v,t} \mathbf{b}_{\mathsf{k},v,t} + \mu_{w,t} \tag{5.6}$$

$$\widetilde{\mathbf{Q}}_t = \mathcal{C}_t \mathbf{B}_{w,t} + \mu_{w,t} \tag{5.7}$$

We take into account the square root kernel [TZK+11] in order to estimate probability density for the local modeling in contrast to multivariate normal distribution as in our previous proposed approach in Chapter 4.

5.5.1 The Objective Function

We formulate the proposed reconstruction methodology as energy minimization problem which is solved with the gradient descent based non-linear optimizer. The objective function includes *prior energy*, *pose energy*, *control energy*, and *smoothness energy*,

$$\widehat{\mathbf{Q}} = \arg\min_{\widetilde{\mathbf{Q}}} (w_{pr} E_{pr} + w_{ps} E_{ps} + w_s E_s + w_c E_c), \tag{5.8}$$

where $\widehat{\mathbf{Q}}$ is the final reconstructed pose and w_{pr}, w_{ps}, w_s and w_c are the user defined weights associated with the energy terms and are computed accordingly. In our experiments, we fix their values as: $w_{pr} = 0.3$, $w_{ps} = 0.2$, $w_s = 0.07$ and $w_c = 0.1$.

5.5.2 Prior Energy Term

This energy term elaborates that how likely the synthesized pose is according to the MoCap priors—the joint angle parameterizations of the $\widehat{\mathsf{K}}_w$nn already exists in the MoCap database. We formulate symmetric square root kernel function $\mathcal{K}$ to estimate probability density as,

$$P \propto \sum_{\mathsf{k}=1}^{\widehat{\mathsf{K}}_w} w_{\mathsf{k},t} \cdot \mathcal{K}\sqrt{|\widetilde{\mathbf{Q}}_t - \mathbf{Q}_{\mathsf{k},t}|^2}, \tag{5.9}$$

where $w_{\mathsf{k},t}$ are the normalized weights associated with $\widehat{\mathsf{K}}_w$ nearest neighbors, $\widetilde{\mathbf{Q}}_t$ is the joint angel parameterizations of the synthesized pose in a PCA space at current frame t and $\mathbf{Q}_{\mathsf{k},t}$ represents the joint angle parameterizations of the retrieved optimal weighted k^{th} $\widehat{\mathsf{K}}_w$ pose. For the energy term, the Equation 5.9 is reformulated as,

$$E_{pr} = \sum_{\mathsf{k}=1}^{\widehat{\mathsf{K}}_w} w_{\mathsf{k},t} \cdot \sqrt{|\widetilde{\mathbf{Q}}_t - \mathbf{Q}_{\mathsf{k},t}|}. \tag{5.10}$$

5.5.3 Pose Energy Term

To impose consistency and acceptability in the reconstructed motion, we introduce the pose energy which minimizes the unwanted artifacts arises due to the 2D-3D transformation and compels the joint positions of the synthesized pose, resulted from the forward kinematics function, according to the prior joint positions of the optimal weighted $\widehat{\mathsf{K}}_w$nn. Mathematically,

$$E_{ps} = \sum_{\mathsf{k}=1}^{\widehat{\mathsf{K}}_w} w_{\mathsf{k},t} \cdot \sqrt{|\widetilde{\mathbf{X}}_t - \mathbf{X}_{\mathsf{k},t}|}, \tag{5.11}$$

where $\widetilde{\mathbf{X}}_t$ is the position vector of the current synthesized pose and the notation $\mathbf{X}_{\mathsf{k},t}$ is the joint positions of the k^{th} $\widehat{\mathsf{K}}_w$nn at frame t.

5.5.4 Smooth Energy Term

In order to avoid the velocity variations as well as the jittering and jerkiness effects, the smoothness energy term is introduced. It imposes the smoothness in a way that the newly reconstructed pose is bound to be according to the previously reconstructed poses at frames $t-1$ and $t-2$, as well as the prior knowledge of the neighboring candidates exists in the MoCap database. Mathematically,

$$E_s = \sum_{\mathsf{k}=1}^{\widehat{\mathsf{K}}_w} w_{\mathsf{k},t} \cdot \sqrt{|\widetilde{\mathbf{U}}_t - \mathbf{U}_{\mathsf{k},t}|}, \tag{5.12}$$

where,

$$\widetilde{\mathbf{U}}_t = \widetilde{\mathbf{Q}}_t - 2\widehat{\mathbf{Q}}_{t-1} + \widehat{\mathbf{Q}}_{t-2}, \tag{5.13}$$

$$\mathbf{U}_t = \mathbf{Q}_t - 2\mathbf{Q}_{t-1} + \mathbf{Q}_{t-2}. \tag{5.14}$$

5.5.5 Control Energy Term

We assume that the MoCap database has the similar samples of the input query motion sequences. Under this assumption, we formulate control energy in two cases like,

Case 1. In first case, we extract the 3D feature sets of the end effectors and the head from the current synthesized pose $\widetilde{\mathbf{X}}_t$ and minimize it with the feature sets of the previous reconstructed pose $\widehat{\mathbf{X}}_{t-1}$ as,

$$E_{c1} = \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} |\widetilde{X}_{i,t} - \widehat{X}_{i,t-1}|}. \tag{5.15}$$

Case 2. In second case, we project the 3D feature sets of the end effectors and the head extracted from the current synthesized pose $\widetilde{\mathbf{X}}_t$ onto the 2D image plane using estimated camera projection matrix $\mathcal{M}_t$ and normalize them. We then minimize distance between the 2D feature sets estimated from the input 2D pose $\mathbf{x}_t$ and the normalized 2D feature sets inferred from the synthesized pose at current frame t as,

$$E_{c2} = \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} |\mathcal{M}_t \widetilde{X}_{i,t} - x_{i,t}|}, \tag{5.16}$$

where $\widetilde{X}_{i,t}$ represents the i^{th} joint's position of the 3D synthesized pose, and $x_{i,t}$ is the i^{th} joint's 2D estimated position at current frame t.

5.6 Results and Analysis

We employ HDM05 [MRC+07] MoCap library which is a heterogeneous dataset with a sampling rate of 120Hz. For our experiments, we first down-sample the MoCap dataset to a sampling rate of 30Hz in order to have the same frame rate with which we have captured the input video control signals. As a result, we get roughly 380K number of 3D human poses. For video input query, we have recorded the motion sequences using Kinect RGB camera with resolution 587×440 pixels and with frame rate 30 frames per second. We deploy the database $\widehat{\mathcal{DB}}_{comp}$ as described in Table 4.1 which includes all motion sequences of the MoCap database HDM05 at sampling rate 30Hz, except those motions which are the part of the test dataset and are given as input query to the developed system. We sample the MoCap dataset with the elevation angles (0–15–90) degrees and the azimuth angles (0–10–350) degrees. We conduct all quantitative and qualitative experiments using this database $\widehat{\mathcal{DB}}_{comp}$. For quantitative comparisons, we measure *average Euclidean distance* in centimeters, between joint positions of the reconstructed pose and the ground truth pose relative to the root joint, Equation (4.13). We evaluate the presented methodology on synthetic examples generated by random camera parameters as well as on the real videos with a variety of motions like straight walking, side walking, walking in a circle, jumping jack and cartwheel motions *etc.* We exploit the same skeleton model with $\mathcal{N}_{\text{j}} = 31$ joints as described in Figure 4.5.

5.6.1 Quantitative Evaluation

We evaluate our proposed methodology quantitatively on synthetic videos which we generate from the MoCap files using some random camera parameters. We develop the testing dataset which consists of the synthetic videos of different motion categories like walking motions (walking 2 steps straight with left/right start, walking 4 steps

Comparisons between different motion classes					
Methods	**Walk Straight**	**Walk in Circle**	**Jumping Jack**	**Cartwheel**	**Average**
[YKW13]	2.573	2.466	7.992	7.605	5.159
Our App. (1)	2.048	2.533	4.314	7.514	4.056
Our App. (2)	**1.873**	**2.341**	**3.508**	**6.614**	**3.631**

Table 5.1: Comparison on different types of motion classes in terms of average reconstruction error (cm), when the test view directions are specified with the azimuth angles 0–5–180 degrees with 5 degree step size and the elevation angles 0–10–90 degrees with 10 degree step size.

Comparisons between different motions at specific views					
Methods	**Walk Straight**	**Walk in Circle**	**Jumping Jack**	**Cartwheel**	**Average**
(i) When view direction is fixed with az = 30 and el = 45.					
[YKW13]	2.059	**1.982**	6.562	**5.859**	4.115
Our App. (1)	1.829	2.245	**4.245**	7.332	3.912
Our App. (2)	**1.760**	2.129	4.626	6.948	**3.865**
(ii) When view direction is fixed with az = 60 and el = 45.					
[YKW13]	2.225	**1.888**	7.480	6.750	4.585
Our App. (1)	1.840	2.175	**4.333**	8.698	4.261
Our App. (2)	**1.710**	2.073	4.403	**6.475**	**3.665**

Table 5.2: Comparison on different types of motion classes in terms of average reconstruction error (cm), when the elevation angle is fixed to 45 degree and **(a)** the azimuth angle is 30 degree, **(b)** the azimuth angle is 60 degree.

straight with left/right start, walking in left/right circle *etc.*), jumping jack motions and the hardest one, the cartwheel motions. We check the performance of our proposed approach with different combinations of experiment scenarios which we discuss in detail one by one as under,

- We first evaluate our approach on the test dataset at every possible test view directions (the azimuth angles (0–5–180) degrees with 5 degree step size and the elevation angles (0–10–90) degrees with 10 degree step size) and the average results are compared with [YKW13] as reported in Table 5.1. From the results, we conclude that our proposed methodology (both cases) especially the *case 2* outperforms for all motion categories. Moreover, our approach executes very good numbers in case of cartwheel motions which are considered as one of the most challenging motions to be reconstructed.

- We further investigate the efficiency of our approach for all motion categories at some specific view directions such as: (i) the view direction with the azimuth angle 30 degree and the elevation angle 45 degree and (ii) the view direction when the azimuth angle is fixed to be 60 degree and the elevation angle 45 degree. The

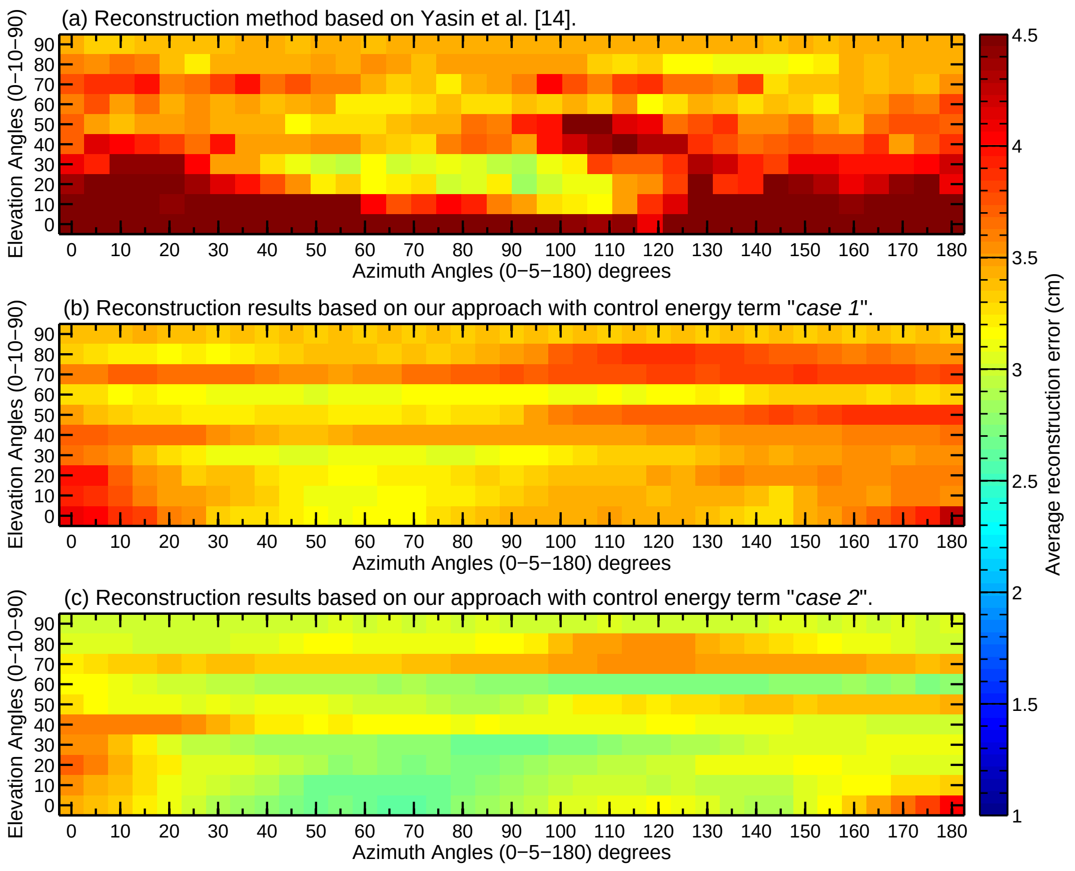


Figure 5.6: Average reconstruction error graph for all kinds of walking motions, with a wide range of test view directions—the azimuth angles (0–5–180) and the elevation angles (0–10–90). **(a)** Yasin *et al.* method [YKW13]. **(b)** Our reconstruction method with control energy term *case 1*. **(c)** Our reconstruction method with control energy term *case 2*.

results for both scenarios are reported in Table 5.2. Our approach with *case 1* executes good results but the *case 2* executes the best reconstruction results on an average as compared to [YKW13].

- We also testify the performance of our approach for every view direction (the azimuth angles = 0–5–180 degrees with 5 degree step size and the elevation angles = 0–10–90 degrees with 10 degree step size) on all types of walking sequences to see the impact of viewpoints on overall reconstruction approach. We report the average reconstruction error for all combinations of azimuth and elevation angles in Figure 5.6, where we construct the *average reconstruction error graphs* with the azimuth view directions along x-axis and the elevation angles along y-axis while the reconstruction error is color-coded. From the results, it is quite obvious that the proposed reconstruction methodology for both cases (whether it is *case 1* or *case 2*) significantly improve the reconstruction results. The developed system

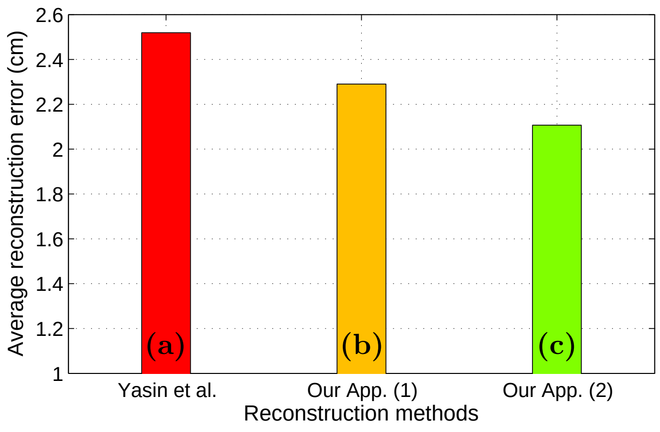


Figure 5.7: The average reconstruction error which is computed by taking average of the reconstructions for all types of walking motion sequences only, at all test view directions—the azimuth angles (0–5–180) and the elevation angles (0–10–90). **(a)** Yasin *et al.* method [YKW13]. **(b)** Our approach with control energy term *case 1*. **(c)** Our approach with control energy term *case 2*.

with *case 2* outperforms and produces the lowest reconstruction error for almost every view direction, as evident from Figure 5.6. At the end, we report the average results on all kinds of walking motions only as shown in Figure 5.7, where again the best results have been executed by our approach *case 2*.

5.6.2 Qualitative Evaluation

We analyse our proposed framework qualitatively on the real monocular video sequences which we have captured through RGB Kinect camera. For real video input query, we first detect and track the video-based feature sets $\mathcal{F}_5^{\text{vid}}$. The ultimate reconstructed sequences depend upon not only the reconstruction methodology but also the fact that how accurately the feature sets are detected and tracked in the given input to prepare the query. For extraction of the feature set from the input video, we rely not only the local feature detectors/descriptors SURF, MSER and colorMSER but also exploits the 3D MoCap priors in the form of $\widehat{\mathsf{K}}$nn as mentioned earlier in Section 5.2. With the use of prior domain knowledge from the MoCap dataset, we are able to tackle the outliers in 2D feature sets, arises due to the occlusions, blurring effects and illuminations *etc.* A few examples for tackling these issues are presented in Figures 5.3–5.4. Figure 5.3 explores that the left hand of the performing actor is completely occluded but with the use of the 3D closest examples ($\widehat{\mathsf{K}}$nn), we are able to find out the position of the occluded hand and track it accurately. Similarly, in Figures 5.4, the hands have lost their structure due to the speed and the blurring effects, and as a result the local feature detectors fail to detect them but with the projection of the 3D MoCap priors we capture their locations precisely. Although we address with occlusion and blurring effects successfully

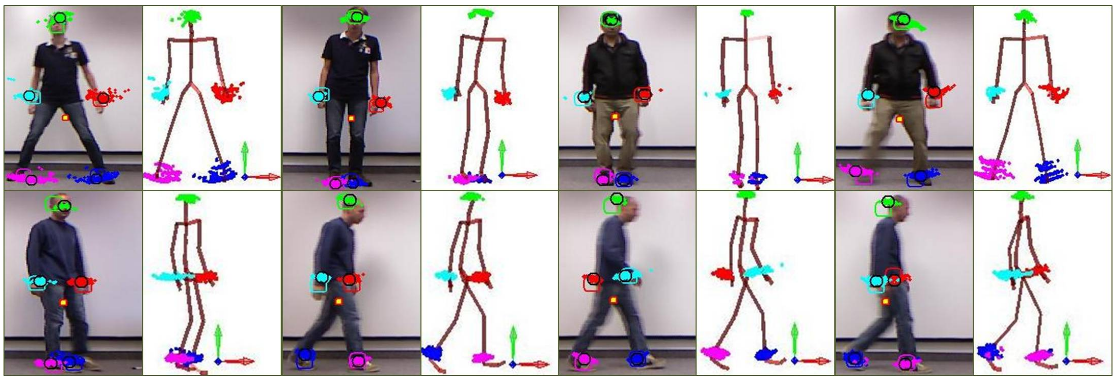


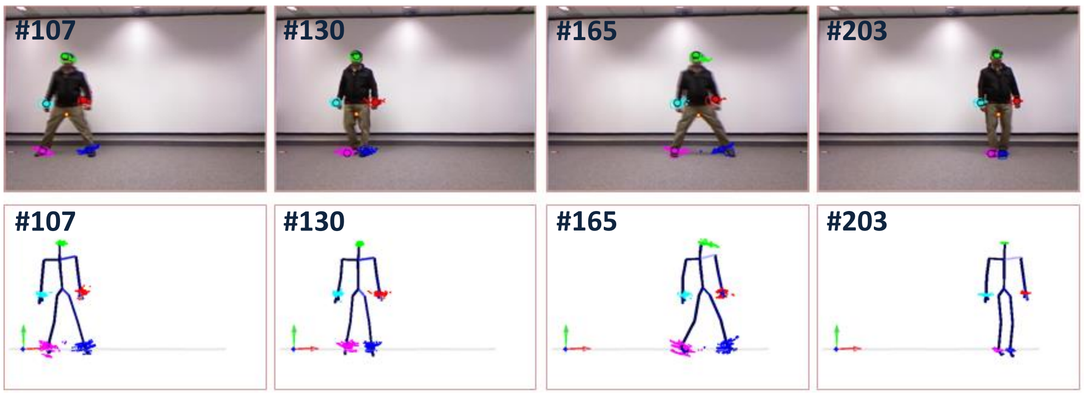
Figure 5.8: A few qualitative results for tracking and reconstruction. The first columns explore the extracted 2D feature set with projected $\widehat{\mathsf{K}}$ nearest neighbors retrieved from the MoCap library while the second columns correspond to the relevant 3D reconstructions, when the presented approach with *case 2* is employed.

but still sometime mistracking of the feature sets may occur which affects the final 3D reconstruction too. The mistracked feature sets are then corrected manually. We have observed from the experiments that the mistracking is roughly 20–25 percent on an average when we do not exploit the 3D MoCap priors, and which is reduced to roughly 7–10 percent on an average by the use of the 3D MoCap priors.

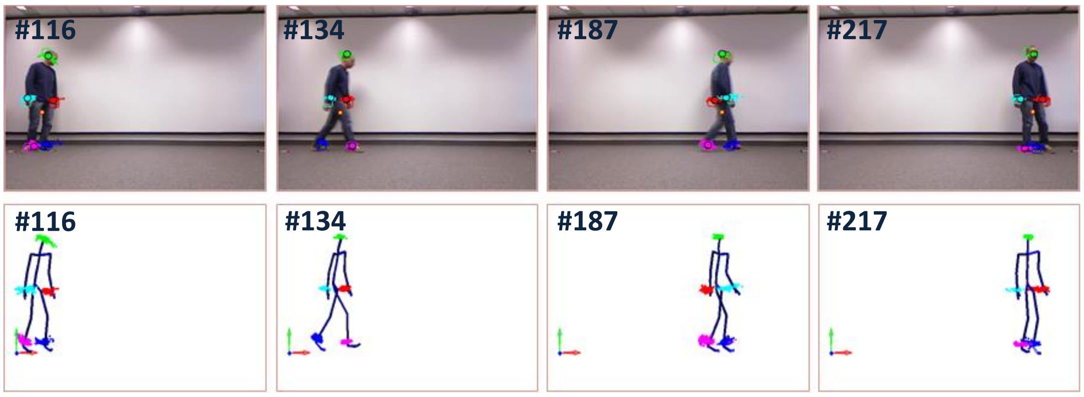
After detection and tracking of the 2D video feature sets, we reconstruct input like 3D motions using the $\widehat{\mathsf{K}}_w$ nearest neighbors. A few qualitative reconstruction results for different types of motion classes are shown in Figure 5.8. Though the improvement in the process of detection and tracking enhances the accuracy in the reconstruction results, but our proposed reconstruction approach is robust enough and produces plausible 3D poses even having some non-anthropometric characteristics in the estimated input 2D pose. This significant property of our approach enables us to handle occlusion and other ambiguities in detection, tracking and the final 3D reconstruction. Some more qualitative tracking and reconstruction results can be seen in Figure 5.9 and Figure 5.10. All these results are executed using the proposed approach with *case 2*.

5.7 Conclusion

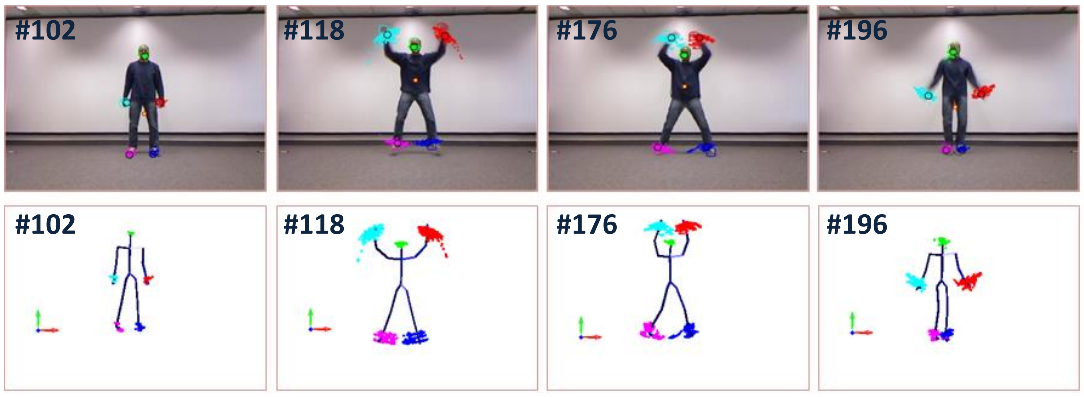
We have proposed an efficient data-driven 3D motion reconstruction approach from the video data by constructing kd-tree data structure and the online lazy neighbourhood graph, taking into account just the positions of the end effectors and the head. The proposed methodology queries for the 2D input control signal and performs the K nearest neighbors search frame-by-frame for 3D prior knowledge available in the MoCap dataset. We then competently utilize the best K closest 3D poses to make the low level feature detection and tracking efficient and more robust. Furthermore, we exploit the weighted optimal closest 3D poses to learn a local 3D pose model as well as to formulate the objective function with multiple energy terms in order to predict the final 3D human motion sequences. We evaluate our methodology quantitative and qualitative on synthetic data as well as on the real monocular video sequences. Our system performs reconstruction with frame rate approximately 5–6 frames per second.



(a) Side-walking-left motion



(b) Walking-left motion



(c) Jumping jack motion

Figure 5.9: Tracking and reconstruction results of our approach *case 2* for different types of motions with projected $\widehat{\mathrm{K}}$nn retrieved from the MoCap library.

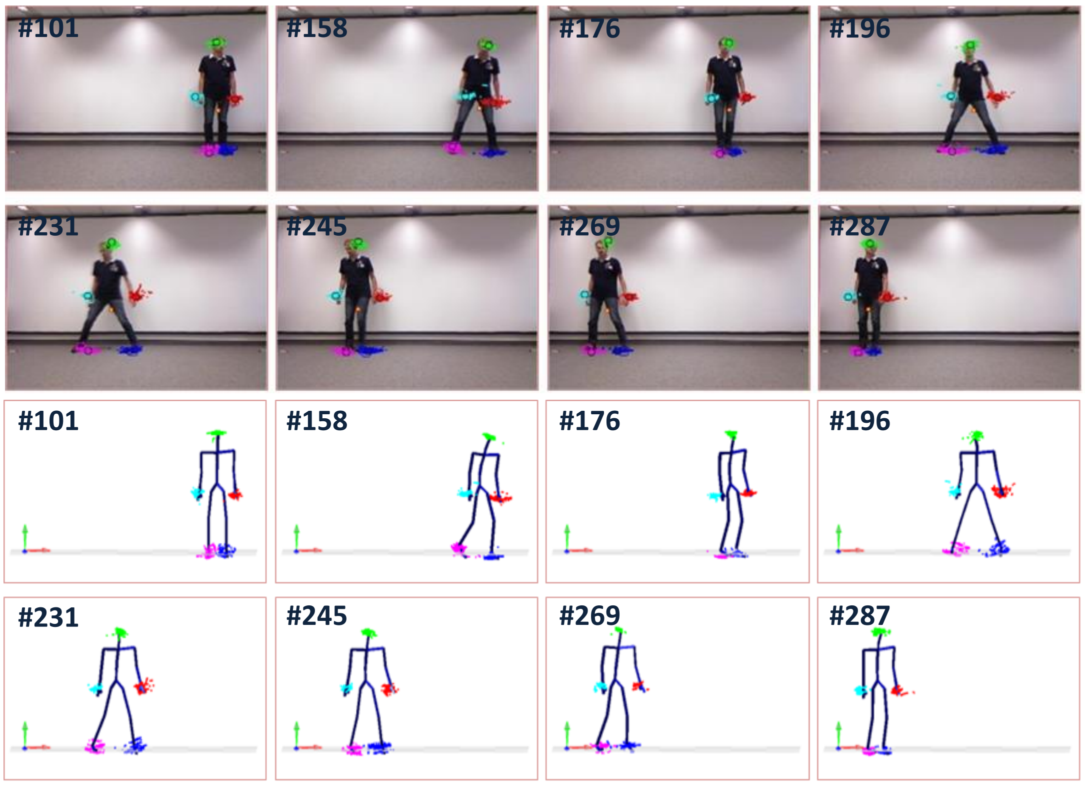


Figure 5.10: Tracking and reconstruction results of our approach *case 2* for side-walking-right motion sequences with projected $\widehat{\mathrm{K}}$nn retrieved from the MoCap library.

Part II

3D Pose Estimation from a Monocular Single Image

6 Recovering 3D Pose from an Image

Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time.

Thomas A. Edison

6.1 Introduction

In this chapter, we deal with the most challenging task of recovering the 3D human pose from just a single monocular image, that may be a synthetic image or a real internet image. The articulated 3D human pose estimation from a monocular single image is a severely under constrained and an ill-posed problem. Given the 2D joint positions in an image, we propose an efficient search & retrieval method for the 3D poses from the motion capture dataset, which leads towards a novel data-driven approach to estimate the final 3D human pose from a monocular still image. For 3D pose retrieval: (i) we design and devise multiple feature sets based on the subsets of 2D joint locations in order to create global similarity search into the MoCap dataset. As we do not rely on any temporal coherence in retrieval and reconstruction processes, therefore we revisit the choice of feature sets consisting of the four end effectors and the head as in previous chapters. Instead, we develop multiple suitable feature sets and the selection is made on their performance evaluations in terms of accuracy, time and memory. (ii) We resolve the 2D-3D cross model retrieval issue efficiently by projecting both, the 3D feature sets derived from the motion capture data, and the 2D feature sets from the image pose, to a normalized 2D pose space. As a result, we retrieve 3D similar poses conveniently from the MoCap dataset just utilizing the subsets of 2D joint locations.

In order to infer the final 3D pose, we apply PCA to reduce dimensionality and compute a 3D pose model from the retrieved K nearest neighbors. We constrain it through pose priors in quaternion pose space $\mathbb{Q}$ as well as in cartesian pose space $\mathbb{R}$ and fit it to the 2D observation by minimizing the projection error. We derive benefits from the joints' weights that are allotted to all joints included in the skeleton model according

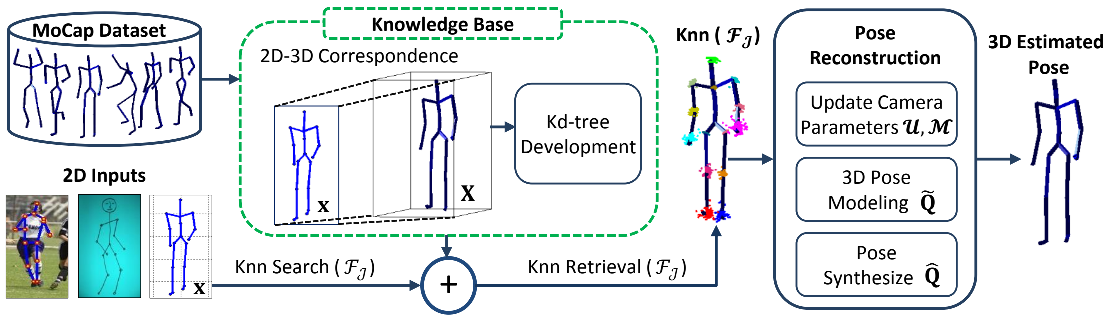
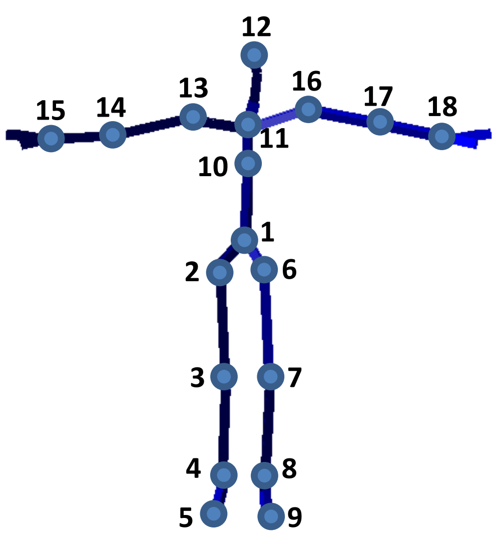

MoCap Dataset
2D Inputs
Knowledge Base
2D-3D Correspondence
Kd-tree Development
Knn ($\mathcal{F}_\mathcal{J}$)
Knn Search ($\mathcal{F}_\mathcal{J}$)
Knn Retrieval ($\mathcal{F}_\mathcal{J}$)
Pose Reconstruction
Update Camera Parameters $\mathcal{U}, \mathcal{M}$
3D Pose Modeling $\tilde{\mathbf{Q}}$
Pose Synthesize $\hat{\mathbf{Q}}$
3D Estimated Pose

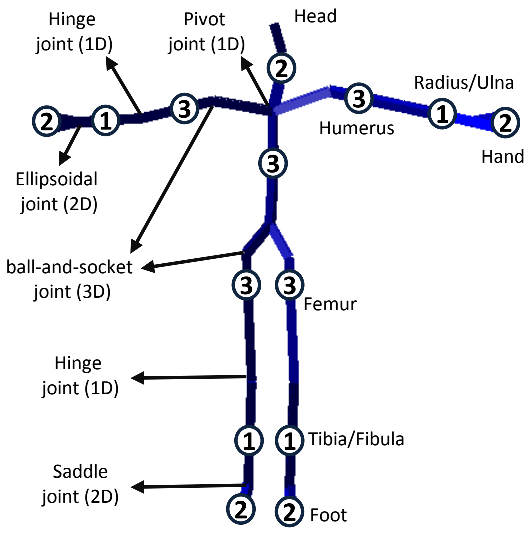
Figure 6.1: System flow diagram. After developing *knowledge base*, we input a single 2D pose extracted either from a synthetic image, a real image or a hand-drawn sketch and create Knn search through kd-tree using 2D feature sets $\mathcal{F}_\mathcal{J} \in \{\mathcal{F}_5^{\text{im}}, \mathcal{F}_7^{\text{im}}, \mathcal{F}_9^{\text{im}}, \mathcal{F}_{11}^{\text{im}}, \mathcal{F}_{14}^{\text{im}}\}$. We optimize and update camera parameters by exploiting these Knn. From the retrieved poses, a 3D pose model is built and fit to the 2D pose in order to reconstruct the final 3D pose through a novel energy function.

to their degree of freedom and as a results we make the 3D pose reconstruction more efficient and robust. Furthermore, we introduce a two-fold system in order to estimate the unknown camera parameters by exploiting the retrieved closest examples as well as the input 2D observations. We first select the suitable viewpoints through sampling of the MoCap data using a simple voting strategy, which are further refined by the retrieved closest examples and by minimizing the deviations from the 2D observations.

We thoroughly evaluate the presented approach quantitatively on a wide range of synthetic 2D images of different activities, which are generated from the MoCap files using some random camera parameters and compare the results with the state-of-the-art approaches. Particularly, we analyze the influence of the skeleton structure discrepancies between the MoCap datasets; the impact of the different feature sets, the camera viewpoints and the noise; and the influence of the skeleton joints in retrieval and reconstruction processes. In our experiments, we show that our approach achieves state-of-the-art results when the test pose is from the same MoCap dataset, but it also achieves competitive results when a completely different MoCap dataset is exploited. We report qualitative analysis of our proposed framework on the real images using PARSE dataset [Ram07], where we utilize off-the-shelf algorithm [YR11] to extract the 2D joint positions from the given monocular image. We also analyze our approach qualitatively on the hand-drawn sketches, where we manually annotate the 2D joint positions in the sketches. We work on the benchmark motion capture datasets like CMU motion capture dataset [CMU14] and HDM05 motion capture dataset [MRC+07], both MoCap datasets are publically available.



(a) Skeleton model



(b) Skeleton with DoF

Joint IDs	Joint weights	Joint IDs	Joint weights	Joint IDs	Joint weights	Joint IDs	Joint weights
1	0.83	6	0.0	11	0.55	16	0.55
2	0.0	7	0.83	12	0.70	17	0.83
3	0.83	8	0.27	13	0.55	18	0.27
4	0.27	9	0.55	14	0.83		
5	0.55	10	0.83	15	0.27		

(c) Joints' weights **w**.

Figure 6.2: **(a)** The skeleton model with all 18 joints. **(b)** The skeleton model shows the joint types with different degree of freedom (DoF), while **(c)** represents the details about the normalized joint weights.

6.2 Overview

We propose an efficient 3D pose retrieval framework as well as an online data-driven 3D pose estimation from a single 2D synthetic/real image. For pose retrieval, we develop a *knowledge base* from the motion capture dataset (Section 4.3.3), which is a preprocessing step and has been performed for once. Given 2D joint locations, we extract the 2D feature sets and perform similarity search from the motion capture database through the *knowledge base*. We design multiple feature sets for the efficient search and retrieval. Section 6.3 provides the details about the search and retrieval of nearest neighbors. In Section 6.4, we explain the two-fold procedure to estimate camera parameters. Having in hand the retrieved nearest neighbors, we construct a 3D pose model on-the-fly and formulate an energy function to fit the model according to the 2D observations in order to reconstruct the 3D pose. We introduce the joint weights that contribute in the energy function which is finally optimized through non-linear gradient descent optimizer. This will be discussed in Section 6.5. In the end, Section 6.6 illustrates the experiments with results and discussions while Section 6.7 consists of the conclusions.

Feature Sets $\mathcal{F}_{\mathcal{J}}$	**Joints Involved in Feature sets $\mathcal{J}_{\mathcal{F}}$**
$\mathcal{F}_5^{\text{im}}$	[4 8 12 15 18]
$\mathcal{F}_7^{\text{im}}$	[4 8 12 13 15 16 18]
$\mathcal{F}_9^{\text{im}}$	[3 4 7 8 12 14 15 17 18]
$\mathcal{F}_{11}^{\text{im}}$	[3 4 7 8 12 13 14 15 16 17 18]
$\mathcal{F}_{14}^{\text{im}}$	[2 3 4 6 7 8 11 12 13 14 15 16 17 18]

Table 6.1: Different types of feature sets which we develop on the basis of the subsets of joints involved in the skeleton model.

6.3 Pose Similarity Search and Retrieval

The major component towards a data-driven reconstruction approach is an efficient similarity search and retrieval from the motion capture dataset. We develop an intermediate container the *knowledge base* as described in Section 4.3.3, through which we make search and retrieval more convenient and robust. We normalize the 3D poses in MoCap dataset by discarding orientation and translation and build up the 3D normalized pose space $\boldsymbol{\Psi}$ which includes the 3D feature sets extracted from the 3D normalized poses and add it into the *knowledge base*.

The input to our reconstruction approach is either a 2D synthetic image generated from the MoCap file by random camera projection like in [FZZW14, RKS12] or some internet real image or a hand-drawn sketch. For 2D query pose, we do not have any cue regarding the joints' orientation, the temporal coherence and the depth knowledge *etc.* Moreover, we have no any camera parameter as well. We perform Knn search into the MoCap dataset just on the basis of the 2D feature sets extracted from the 2D pose with missing degree of freedom. As a solution, we construct the 2D normalized pose space ψ by projecting the 3D normalized feature sets onto the 2D image plane using orthographic projection through a several virtual cameras with a number of view directions in the form of the azimuth angles and the elevation angles. We create 24×7 virtual cameras by spanning the azimuth angles (0–15–345) degrees and the elevation angles (0–15–90) degrees, both with step size 15 degree. We will discuss the influence of virtual cameras in Section 6.6.3. For an appropriate 2D-3D matching and correspondence, we rescale both the 3D normalized poses as well as the 2D normalized poses with an arbitrary scaling factor. To this end, we have 2D normalized pose space ψ which we also include into the *knowledge base*.

We devise multiple feature sets that consist of different joint combinations and hold the proper skeleton characteristics. These feature sets with the relevant subsets of joints are reported in Table 6.1 and the details about these joints in the skeleton model are

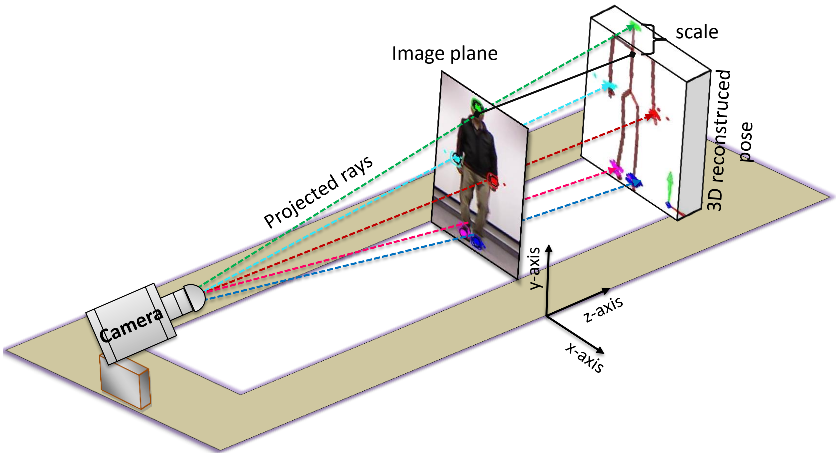


Figure 6.3: The weak perspective camera model where the depth information has been recovered through the prior information embedded in the MoCap dataset. This exemplary illustration is for the feature sets $\mathcal{F}_5^{\text{im}}$.

illustrated in Figure 6.2. We develop these feature sets on the basis of joints' creditabilities in terms of robust similarity search and retrieval. Our main objective is to retrieve the robust closest 3D poses from the MoCap dataset efficiently without consuming enormous memory and time. In [YKW13, KTWZ10], the authors argue that most worthy joints are the four end effectors (the left/right hands and the left/right feet) and the head. As we do not consider any temporal information, we combine a few more joints in addition to these end effectors and develop a set of feature sets as described in Table 6.1 and evaluate their performance in several ways (see Section 6.6.3).

From the given 2D query pose, we derive the 2D feature sets based on the different 2D joint positions. Earlier than the query composed of these 2D feature sets is given as input to the system for similarity search and retrieval, we normalize the 2D feature sets by transforming all joints to its root node, the center of the mass, and rescale them according to the fixed arbitrary scale level. To this end, we have 2D feature sets, either existing in the 2D normalized pose space in *knowledge base* or extracting from the 2D query pose, both has become comparable in order to search and retrieve the closest examples from the MoCap dataset efficiently. We use a kd-tree for fast search and retrieval of nearest neighbors [YKW13, TZK$^+$11, KTWZ10].

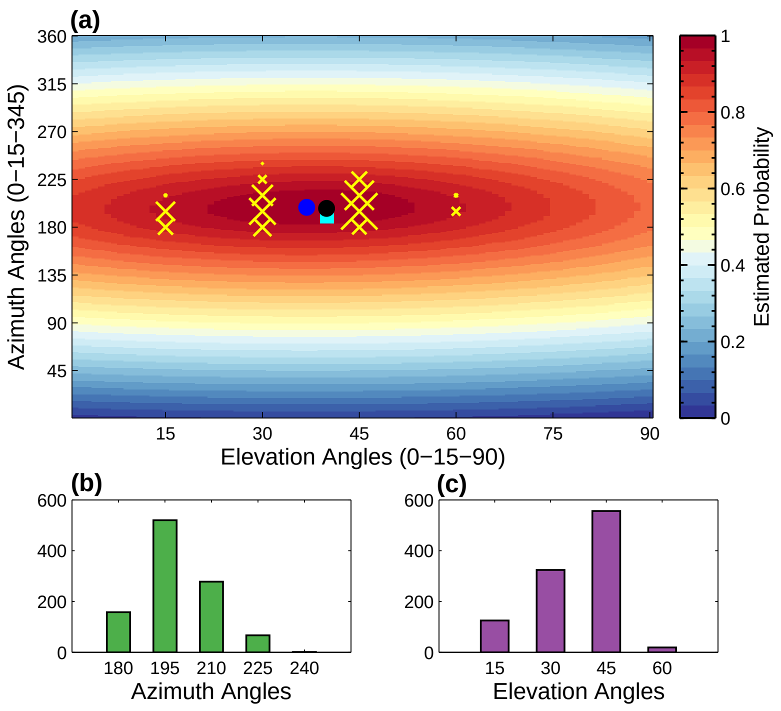


Figure 6.4: **(a)** An estimation of the camera view directions. The **yellow cross** (×) symbols represent the clusters of the view directions that we observe by the nearest neighbors retrieved through the sampling of the MoCap data at various azimuth and elevation angles. Bigger cross (×) symbol indicates a bigger cluster (the more nearest neighbors are retrieved through this specific viewpoint) as compared to the smaller cross (×) symbol. We compare the camera viewpoint results obtained through the symmetric square root kernel approach against the results when we compute the simple arithmetic mean of all view directions observed through the retrieved nearest neighbors. The **dark black circle** represents the results of square root kernel function; **blue circle** shows the mean value, and **cyan square** shows the ground truth view directions. Figures **(b)** and **(c)** represent the histograms of the azimuth angles and the elevation angles respectively. For this experiment, we employ 1024 nearest neighbors.

6.4 Camera Parameters

In this paper, we consider the weak perspective camera model. For this camera model, the projection matrix $\mathcal{M}$ is defined as,

$$\mathcal{M} = \begin{bmatrix} s_x & & \\ & s_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} r_1^T & T_1 \\ r_2^T & T_2 \\ 0^T & 1 \end{bmatrix} \tag{6.1}$$

The notations s_x and s_y are the scaling factors along x-axis and y-axis, r_1 and r_2 denote the rotation parameters while T_1 and T_2 represent the translation vector. An exemplary illustration of camera model has been shown in Figure 6.3. We consider the translation zero under the assumption that the 3D center of the mass coincides the 2D centroid. The projection matrix $\mathcal{M}$ is the weak perspective projection matrix and is characterized by the first two rows orthogonal to each other. This formulation is similar to [WWL+14, FZZW14, RKS12].

For estimation of the camera parameters, we adopt the two-fold nonlinear optimization method. To this end, we minimize the energy for the camera estimation as,

$$\mathsf{e}_c = \arg\min_{\mathcal{U},\mathcal{M}}(a\mathsf{e}_g + b\mathsf{e}_p), \tag{6.2}$$

where $\mathcal{U}$ is a vector that represents camera view directions with azimuth and elevation angles. The notations e_g and e_p are the energy terms with the associated weights a and b having values 0.45 and 0.55 respectively. In **first phase**, we predict the camera view directions in the form of azimuth and elevation angles through the sampling of the MoCap data at different view directions. We formulate view direction estimation as the multi-label classification task with 24×7 number of classes corresponding to the virtual cameras which are basically the viewpoints with different combinations of azimuth and elevation angles (see Section 6.3). We exploit the retrieved nearest neighbors voting observations in order to predict the camera view directions. Each nearest neighbor is labeled positive for that class to which it belongs. For K nearest neighbors, we obtain a number of voting clusters for virtual cameras with azimuth and elevation angles, as shown with **yellow cross** ($\times$) symbols in Figure 6.4 (a). The higher votes for a specific class of virtual camera result into a bigger cluster that is represented by the bigger **yellow cross** ($\times$) symbol. We elaborate the clustering results more precisely with histograms which we develop separately for azimuth and elevation angles as shown in Figure 6.4 (b)-(c). To this end, we have primary camera viewpoints which are the initial points for the final camera parameter estimation. We optimize these clusters with the square root kernel function to obtain the optimal camera viewpoints as,

$$\mathsf{e}_g(\mathcal{U}) = \sum_{\mathsf{k}=1}^{\mathsf{K}} \cdot \sqrt{\|\mathcal{U} - \mathcal{V}_{\mathsf{k}}\|}, \tag{6.3}$$

where $\mathcal{V}_{\mathsf{k}}$ represents the k^{th} viewpoint observed during nearest neighbors retrieval process.

In **second phase**, we not only refine the camera viewpoints but also estimate the remaining camera parameters. Given the joint positions of the 2D pose, the K-similar 3D poses and the initial camera view directions, we refine and upgrade the camera parameters through energy as,

$$\mathsf{e}_p(\mathcal{M}) = \sum_{\mathsf{k}=1}^{\mathsf{K}} \cdot \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} \|\mathcal{M} \cdot X_{i,\mathsf{k}} \; - x_i\|}, \tag{6.4}$$

where x_i is the i^{th} 2D joint position and $X_{i,\mathsf{k}}$ represents the i^{th} 3D joint position of the k^{th} nearest neighbor.

We use the square root as a symmetric kernel function during optimization. Such a kernel based representation is well suited to approximate the arbitrary shaped probability density including multiple peaks [TZK+11]. Multiple peaks can especially occur when multiple clusters of nearest neighbors are found as evident from Figure 6.4, where it is quite obvious that even having multiple clusters, we are able to get more robust camera viewpoints with the square root kernel function as compared to the simple arithmetic mean. The two-fold nonlinear optimization method allows to get very good initializations of camera viewpoints from the first phase, which are further refined in the second phase. Good initializations of camera viewpoints are essential to estimate more accurate camera parameters, to speed up the optimization and to infer the plausible 3D pose. The camera viewpoints $\mathcal{U}$ and the final camera projection matrix $\mathcal{M}$, both are found by Levenberg-Marquardt optimization algorithm using nonlinear optimizer.

6.5 Pose Reconstruction

In order to reconstruct the final 3D pose, we compute a linear pose model using principal components analysis from the retrieved poses $\mathcal{Q} = \{\mathbf{Q}_1, \ldots, \mathbf{Q}_{\mathsf{K}}\}$,

$$\widetilde{\mathbf{Q}} = \mathcal{C}\mathbf{B} + \mu, \tag{6.5}$$

and fit it to the image. Recall that the term $\mathbf{B}$ represents a set of basis vectors, μ is the mean of the Knn and $\mathcal{C}$ is the current pose in coordinates of PCA space. To this end, we minimize the energy,

$$\widehat{\mathbf{Q}} = \arg\min_{\widetilde{\mathbf{Q}}}(\omega_p E_p + \omega_c E_c), \tag{6.6}$$

using gradient based optimizer. The notations ω_p and ω_c are the corresponding energy weights. The first term E_p measures the deviation from the retrieved poses while the second term E_c measures the projection error with respect to the input 2D query pose. We introduce weight for each joint included in the skeleton model according to the degree of freedom. We assume that the joints with higher degree of freedom have the greater influence upon the movements of the body parts as compared to the joints with lower degree of freedom. In this context, we allocate higher weights to the joints with higher degree of freedom *e.g.*, the ball-and-socket joints with 3 degrees of freedom have higher weights as compared to the hinge joints with 1 degree of freedom. We then normalize these allocated weights and denote them with a vector, $\mathbf{w} = \{w_1, \ldots, w_{\mathcal{J}}\}$. The details about skeleton model and the different joint types with their degree of freedom have been shown in Figure 6.2(a) and Figure 6.2(b) respectively, while the joint weights $\mathbf{w}$ are reported in Figure 6.2(c).

6.5.1 Retrieved Pose Error

For a large heterogeneous dataset, the human pose synthesize in low dimensional space just on the basis of joint angle configurations, when the query is composed of only a subset of 2D joint positions, does not produce plausible results. It may produce some jittering or unwanted artifacts in final 3D pose. For that reason, we penalize the deviation not only from the joint angle parameterizations of the retrieved poses in the quaternion pose space $\mathbb{Q}$ but also the joint positions of the retrieved poses in the cartesian pose space $\mathbb{R}$,

$$E_p = \omega_{pa} E_{pa} + \omega_{pp} E_{pp}, \tag{6.7}$$

with corresponding weights ω_{pa} and ω_{pp}. For E_p, we consider all joints included in the skeleton model whether these joints take part in the retrieval process or not.

E_{pa} enforces the synthesized pose $\widetilde{\mathbf{Q}}$ according to the prior knowledge in the form of joint angle parameterizations already exists in the MoCap dataset,

$$E_{pa}(\widetilde{\mathbf{Q}}) = \sum_{\mathsf{k}=1}^{\mathsf{K}} \cdot \sqrt{\sum_{i \in \mathcal{J}} \| w_i \cdot (Q_{i,\mathsf{k}} - \widetilde{Q}_i) \|}, \tag{6.8}$$

where w is the weight for each joint.

E_{pp} directly constrains the 3D joint positions of the synthesize pose from the retrieved similar poses in the cartesian pose space $\mathbb{R}$,

$$E_{pp}(\widetilde{\mathbf{Q}}) = \sum_{\mathsf{k}=1}^{\mathsf{K}} \cdot \sqrt{\sum_{i \in \mathcal{J}} \| w_i \cdot (X_{i,\mathsf{k}} - \mathbf{f}(\widetilde{Q}_i, \mathcal{S}_i)) \|}, \tag{6.9}$$

where $\mathbf{f}$ is the forward kinematics function that computes joint positions $\widetilde{\mathbf{X}}$ from the joint angle configurations of the synthesized pose $\widetilde{\mathbf{Q}}$ while $\mathbf{S}$ is the skeleton model which we have developed by taking average of all subjects' skeletons that are included in the MoCap dataset. The term $X_{i,\mathsf{k}}$ is the i^{th} joint position of the retrieved k^{th} nearest neighbor.

Databases	MoCap Datasets	Details
$\mathcal{DB}_{cmu}$	CMU	It contains the 3D poses of CMU MoCap dataset except all those poses from which we generate the 2D synthetic input test dataset. Furthermore, we also remove all those motion sequences completely from which even a single 2D synthetic input image is generated so that the database become totaly free from overlapping with any testing image.
$\overline{\mathcal{DB}}_{cmu}$	CMU	It contains the 3D poses of CMU MoCap dataset and we remove all motion sequences completely, to which even a single 2D synthetic input pose belongs, as well as all those motion sequences which are performed by the same performing actor present in the 2D synthetic input image.
$\mathcal{DB}_{hdm}$	HDM05	This database is developed using HDM05 motion capture sequences.

Table 6.2: The details of different databases which are developed for the evaluation of the proposed system.

6.5.2 Projection Error

For measuring the deviation from the 2D input pose, we use the inferred projection matrix $\mathcal{M}$ (Section 6.4) to project the model onto the given query image. The projection error is then given by,

$$E_c(\widetilde{\mathbf{Q}}) = \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} \|w_i \cdot (\mathcal{M} \cdot \mathbf{f}(\widetilde{Q}_i, \mathcal{S}_i) - x_i))\|}. \tag{6.10}$$

In computing projection error, we regard only those joints which involve in developing the specific feature sets and are exploited for the search and retrieval of Knn, $\mathcal{J}_{\mathcal{F}}$.

6.6 Experiments

We evaluate the developed system's performance on two types of test input data, the synthetic 2D images as well as the internet real images. We develop the synthetic 2D image dataset from the 3D MoCap files by random camera parameters [FZZW14, RKS12]. The skeleton model consists of 18 joints including head, neck, chest, root, left/right shoulders, left/right elbows, left/right wrists, left/right hips, left/right knees, left/right ankles and left/right feet (see Figure 6.2). We follow the same protocol [FZZW14] for

Input Dataset						
Items	**Walking**	**Running**	**Jumping**	**Boxing**	**Climbing**	**Total**
No. of Poses	13509	2970	5913	9128	12289	43809
No. of Subjects	8	8	4	4	1	25

Table 6.3: The details of the synthetic 2D image test dataset $\mathcal{TS}_1$ generated from the CMU MoCap files by some random camera parameters. This test dataset is used for quantitative evaluation.

the performance evaluation in terms of error measurement such as, the normalized *reconstruction error* and the *reconstruction rate*. For normalized *reconstruction error*, we first compute the Euclidian distance between each joint of the estimated 3D pose and the ground truth 3D pose and select the joint with **maximum** error. The error is measured then by the fraction of the backbone length in order to fix the arbitrary scale for the different skeletons. For multiple images, we take average of the *reconstruction error* and refer it as *average reconstruction error*. The *reconstruction rate* is computed by taking the percentage of those test images which contain very low normalized *reconstruction error* subject to some threshold level. We select the same threshold level 0.3 in the line of [FZZW14]. We refer the *average reconstruction error* shortly as *rec_err* and the *reconstruction rate* as *rec_rate*. Like previous state-of-the-art approaches [FZZW14, RKS12], we Procrustes align the reconstructed pose with the ground truth before calculating the error.

6.6.1 Datasets

MoCap Datasets. We employ two different motion capture datasets in order to evaluate our approach, CMU motion capture dataset [CMU14] and HDM05 motion capture dataset [MRC+07]. Both datasets are publicly available. For CMU MoCap dataset, due to limited memory capacity, we pick up roughly $^1/_3$ of CMU MoCap dataset for developing the *knowledge base*, which consists of a variety of motion classes like walking, running, jogging, step walking, kicking, punching, lifting up, jumping, and other locomotion and sports activities in order to justify the generalizability of our proposed approach. For HDM05 dataset, we employ all motion capture files available in the dataset. We first down-sample our MoCap datasets from 120Hz to 30Hz. As a result, we get roughly 360K frames for CMU dataset and 380K frames for HDM05 dataset. For quantitative evaluation, we develop the databases according to three different experiment scenarios such as, $\mathcal{DB}_{cmu}$, $\overline{\mathcal{DB}}_{cmu}$ and $\mathcal{DB}_{hdm}$ as reported in Table 6.2.

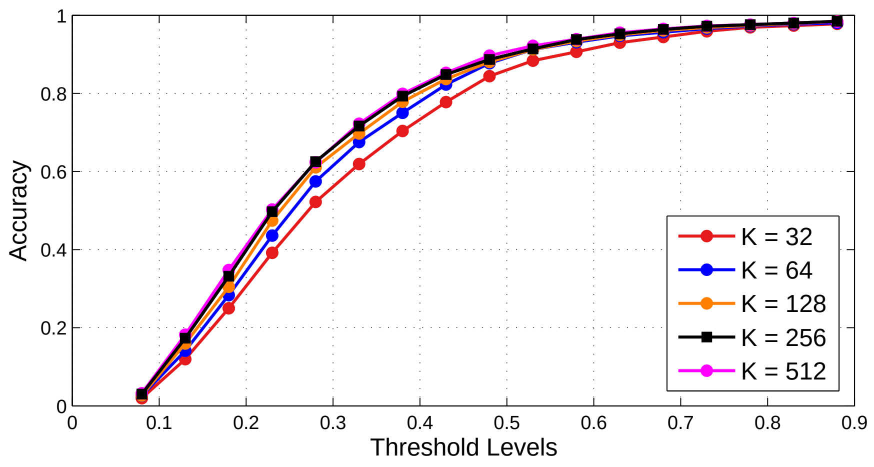


Figure 6.5: Comparison between different numbers of nearest neighbors with respect to average reconstruction error *rec_err* at various threshold levels. We carry out this experiment on the test dataset $\mathcal{TS}_2$.

Input Datasets. Similar to [FZZW14, RKS12], we generate the synthetic 2D input image test datasets from the motion capture files of CMU MoCap dataset by some random camera parameters for quantitative analysis. We select those action categories as in [FZZW14] like, walking, running, jumping, boxing and climbing. Our synthetic input test dataset, referred as $\mathcal{TS}_1$, consists of 43809 numbers of 2D synthetic test images from 25 subjects (for detail see Table 6.3) that is large enough as compared to [FZZW14] which uses $29,336$ synthetic images from 23 subjects. We also develop a mini test dataset $\mathcal{TS}_2$ which is the subset of the test dataset $\mathcal{TS}_1$ and consists of 3500 2D synthetic images. We randomly select these 3500 synthetic images from all action classes such as, walking, running, jumping, boxing and climbing that are included in our test dataset $\mathcal{TS}_1$. On this test dataset $\mathcal{TS}_2$, we perform some pre-experiments in order to tune the parameters. We also carry out a few sub-experiments utilizing this mini test dataset. For qualitative analysis, we employ PARSE dataset [Ram07] which contains internet real images. We also evaluate the robustness of the proposed approach on the hand-drawn sketches which we draw for a few action classes like walking, jumping jack and grabbing.

6.6.2 Parameters

Nearest Neighbors. We first conduct an experiment to find out that how many nearest neighbors should be enough to our proposed system in order to reconstruct robust and plausible 3D pose. We also illustrate that how the different numbers of nearest neighbors exert influence on the overall reconstruction approach at different threshold levels. We fix the values for K such as 32, 64, 128, 256 and 512 and evaluate

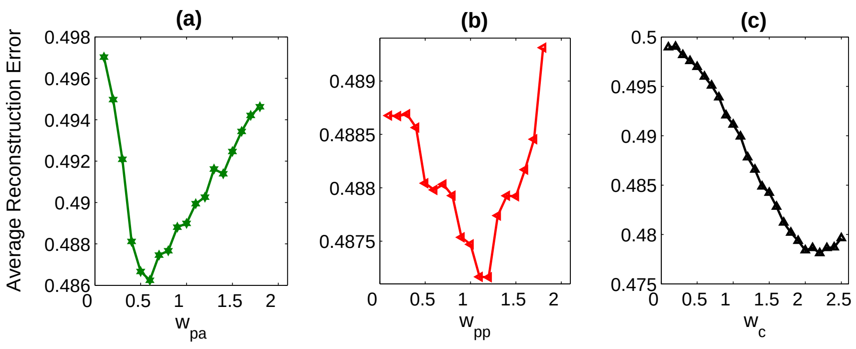


Figure 6.6: Impacts of weighted energy terms, E_{pa}, E_{pp} and E_c are shown in **(a)**, **(b)** and **(c)** respectively in terms of average reconstruction error *rec_err*.

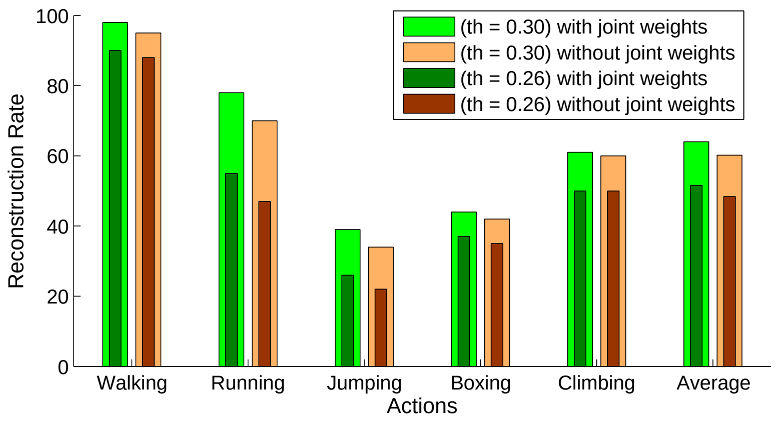


Figure 6.7: The impact of the joint weights **w**. We conduct this experiment on $\mathcal{TS}_2$ and compute reconstruction error rate *rec_rate* at threshold levels 0.3 and 0.26 respectively.

the performance of our approach in terms of accuracy rate at different threshold levels. We have found from the results that when the size of nearest neighbors equals to 256, the system executes more accurate reconstructions comparatively at almost all threshold levels. By increasing the number of nearest neighbour like $\mathsf{K} = 512$, the presented system does not improve the results significantly as evident from Figure 6.5 where the results impose that $\mathsf{K} = 256$ is sufficient to our system to produce the best reconstruction results. We set this value $\mathsf{K} = 256$ for all our experiments independent of that whatever the dataset we deploy in our experiments. We conduct this experiment on the test dataset $\mathcal{TS}_2$.

Energy Weights. We use different energies, E_{pa}, E_{pp} and E_c in our reconstruction approach (Section 6.5). We examine the impact of these energies on the reconstruction results by adjusting different weights for these energies. We allot different weights for an energy starting from 0 while the weights for other energies are kept fixed to their specific values. The results are reported in Figure 6.6 (a)-(c), which show that these energy terms contribute significantly in dropping the error for 3D pose estimation. We adjust the weights as: $\omega_{pa} = 0.8$, $\omega_{pp} = 1.4$ and $\omega_c = 1.8$ for all further experiments.

Joint Weights. In order to testify the impact of the joint weights on the presented system, we carry out an experiment on test dataset $\mathcal{TS}_2$ and compute the reconstruction error rate *rec_rate*, when the threshold levels are fixed to 0.3 and 0.26 respectively. We have found from the results reported in Figure 6.7 that the accuracy rate *rec_rate* drops significantly for all action categories on both threshold levels when we do not make use of the joint weights $\mathbf{w}$ in the proposed reconstruction methodology. The details about the joint weights can be seen in Figure 6.2(c).

6.6.3 Search and Retrieval

We evaluate the proposed search and retrieval framework by conducting different experiments. We first analyse the designed feature sets thoroughly in terms of similarity measure, accuracy, memory consumption and time. We then investigate that how the virtual cameras exert influence on the overall retrieval and reconstruction framework. We perform these experiments on the test dataset $\mathcal{TS}_2$.

Feature Sets. We develop a variety of feature sets on the basis of the subsets of the joints as illustrated in Table 6.1 to perform similarity search into the MoCap dataset (Section 6.3). We examine these feature sets in different ways as,

- In **first** experiment, we check the efficiency of the developed feature sets with respect to similarity measure (the retrieval of the similar poses from the MoCap dataset), with fixed value of $\mathsf{K} = 256$. For this experiment, we select 1500 random images which results into 256×1500 number of retrieved poses for each feature set. We then compute the similarity measure for the retrieved nearest neighbors—the total number of retrieved similar poses that are more close to the ground truth and yield the error in the form of average reconstruction error *rec_err* less than some specific threshold level. We observe from the results reported in Figure 6.8 that the feature set $\mathcal{F}^{\mathrm{im}}_{11}$ retrieves a good number of very similar poses comparatively for almost all threshold levels.

- In **second** experiment, we testify the performance of the feature sets with respect to the final reconstruction for different action classes. We compute the average reconstruction error *rec_err* for the evaluation. The experimental results shown in Figure 6.9 execute that the feature sets $\mathcal{F}^{\mathrm{im}}_{11}$ and $\mathcal{F}^{\mathrm{im}}_{14}$, both produce lower average

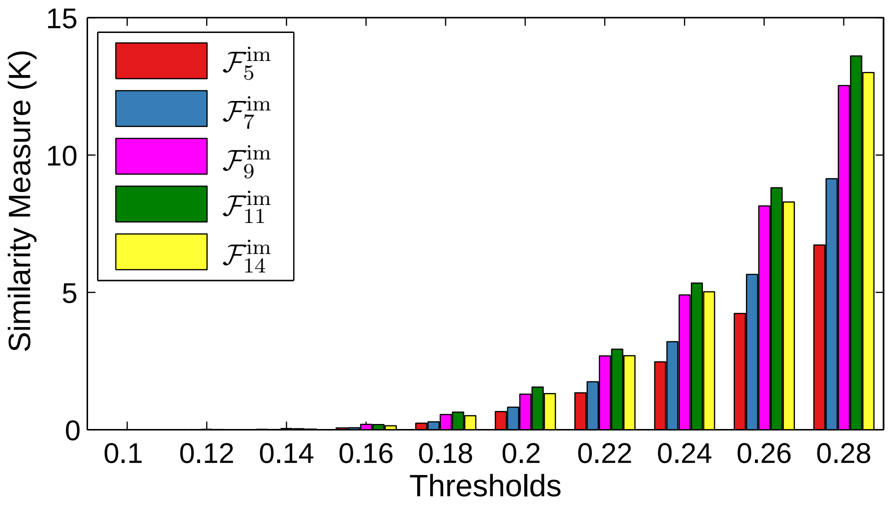


Figure 6.8: The comparison between all developed feature sets on the basis of similarity measure—the total number of the retrieved similar poses from the MoCap dataset that are more close to the ground truth, computed through average reconstruction error *rec_err* under constraints of different threshold levels, when the number of nearest neighbors are fixed to be 256. We perform this experiment on 1500 synthetic images which are selected randomly and as a result it becomes 256×1500 target poses.

reconstruction errors for all action classes as well as on an average over all these action classes, as compared to the other feature sets.

- In **third** experiment, we evaluate the feature sets on the basis of time consumption and memory allocation. The results as reported in Table 6.4 elaborate that although the feature set $\mathcal{F}^{\text{im}}_{5}$ consume less time comparatively in retrieval and reconstruction process but this time difference is not so much critical. Moreover, the time difference for developing the *knowledge base* and the kd-tree can be ignored and does not matter a lot due to the reason that both are the preprocessing steps and need to be performed just only for once.

  In case of memory allocation, the feature sets with more joints allocate more memory comparatively *e.g.*, the feature set $\mathcal{F}^{\text{im}}_{14}$ requires more memory than the feature set $\mathcal{F}^{\text{im}}_{11}$.

From these experiments, we conclude that in the context of accuracy, time consumption and memory allocation, the feature set $\mathcal{F}^{\text{im}}_{11}$ is the best choice in contrast to [YKW13, KTWZ10] where the authors recommend the feature sets $\mathcal{F}^{\text{im}}_{5}$ but they employ the temporal coherence in the retrieval and reconstruction processes. No doubt, $\mathcal{F}^{\text{im}}_{5}$ consumes less time and memory but the accuracy drops significantly in our case when we do not consider the temporal information at all. Generally, the selection of the suitable feature sets is a trade-off between the accuracy and the time & memory consumption. We select

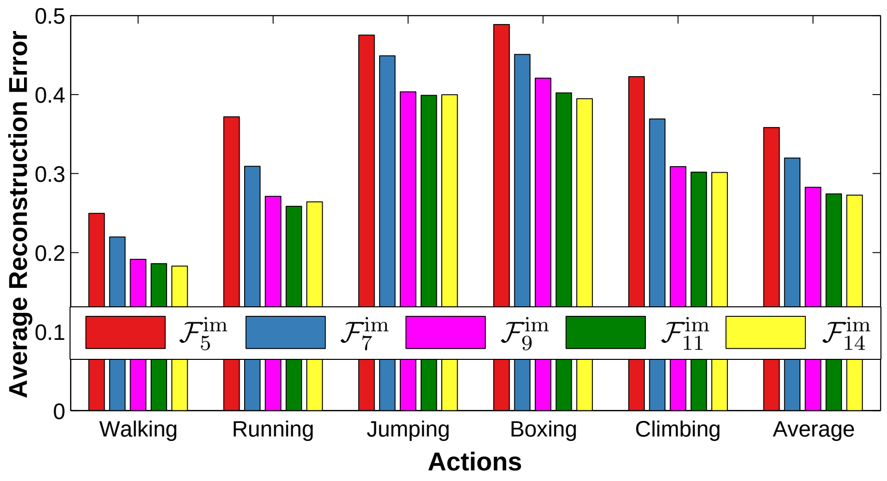


Figure 6.9: Comparison between the feature sets in terms of average reconstruction error *rec_err* on different action categories.

Execution Time (sec.) for Feature Sets					
Components	$\mathcal{F}_5^{\text{im}}$	$\mathcal{F}_7^{\text{im}}$	$\mathcal{F}_9^{\text{im}}$	$\mathcal{F}_{11}^{\text{im}}$	$\mathcal{F}_{14}^{\text{im}}$
(a) Knowledge base	30.56	42.14	54.80	67.89	77.67
(b) kd-tree	97.61	118.12	130.27	144.49	197.74
(c) Retrieval & Reconstruction	0.55	0.57	0.63	0.68	0.97

Table 6.4: The execution time in **seconds** for different feature sets. **(a)** The time that is consumed to develop the *knowledge base* including the creation of the virtual cameras through database sampling. **(b)** The time which is required to develop kd-tree, while **(c)** explores the time for the retrieval and reconstruction process including the time for camera parameters estimation in seconds per image. This execution time is calculated on CMU dataset with 360K number of frames and using 24×7 number of virtual cameras. Note that both **(a)** and **(b)** are the execution times for the pre-processing steps.

the feature set $\mathcal{F}_{11}^{\text{im}}$ and continue our experiments utilizing this feature set $\mathcal{F}_{11}^{\text{im}}$ which is more accurate and consumes very acceptable time and memory comparatively.

Virtual Cameras. We create a number of virtual cameras by sampling of the MoCap database at different azimuth and elevation angles to resolve the 2D-3D correspondence issue (see Section 6.3). To investigate the overall impact of these virtual cameras, we evaluate our approach by designing a variety of database sampling compositions in order to create virtual cameras. The results in Figure 6.10 show the benefits of the use of the virtual cameras. We found that when we increase step size for azimuth angles to 15, 25, 35, 45, and 60 degrees, the error increases correspondingly as well. Similar behaviour is observed for the elevation angles. In short, when we employ more virtual cameras with

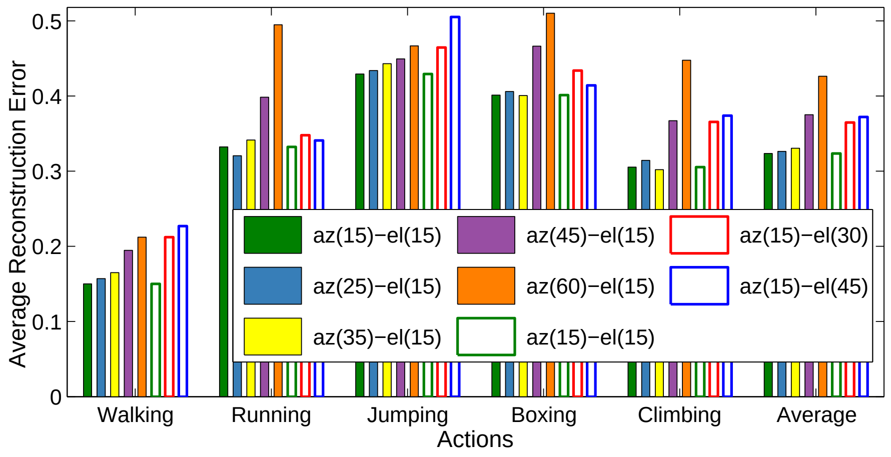


Figure 6.10: Influence of the virtual cameras, created through database sampling at different viewpoints (azimuth and elevation angles), with respect to average reconstruction error *rec_err*. We first fix the step size for the elevation angle to 15 degree and vary the step size for the azimuth angles to 15, 25, 35, 45, and 60 degrees to evaluate the influence of sampling on the basis of the azimuth angles. For elevation angles, we fix the step size for the azimuth angle to 15 degree and step size for the elevation angles changes to 15, 30 and 45 degrees. The filled colored boxes correspond to the sampling on the basis of the azimuth angles while the unfilled colored boxes correspond to the sampling on the basis of the elevation angles.

azimuth or elevation angles, the average reconstruction error decreases correspondingly.

6.6.4 Quantitative Evaluation on Synthetic Images

We perform quantitative analysis of our proposed methodology on the test dataset $\mathcal{TS}_1$ which consists of 43809 synthetic 2D images (Table 6.3) and compare the results with the state-of-the-art approaches [FZZW14, RKS12]. For evaluation, we design different experimental setups on the basis of the MoCap datasets as reported in Table 6.2.

- In **first** case ($\mathcal{DB}_{cmu}$), when we employ CMU MoCap dataset as a prior and remove all the sequences of that motion capture clip from the dataset, from which we generate even a single synthetic 2D input image so that we can avoid any overfitting. We report the results in Table 6.5(a) which elaborate that our approach outperforms the other state-of-the-art methods [FZZW14, RKS12] for all five activities in terms of normalized *average reconstruction error* as well as *reconstruction rate* with 0.3 threshold level.
- For **second** case ($\overline{\mathcal{DB}}_{cmu}$), when we use CMU MoCap dataset and remove not only the all motion sequences containing synthetic 2D input image but also remove the all motion sequences performed by the same actor present in 2D input image. The results as presented in Table 6.5(b) show that our methodology again performs

Comparison with state-of-the-arts							
Methods	**Err. Metrics**	**Walking**	**Running**	**Jumping**	**Boxing**	**Climbing**	**Average**
[FZZW14]	*rec_err*	0.260	0.385	0.316	0.530	0.526	0.403
	rec_rate	73.9%	38.2%	41.6%	17.0%	28.1%	39.8%
[RKS12]	*rec_err*	0.446	0.453	0.374	0.584	0.533	0.478
	rec_rate	29.6%	23.0%	31.6%	10.7%	20.1%	23.0%
(a) Results with $\mathcal{DB}_{cmu}$ (MoCap from CMU dataset)							
Our App.	*rec_err*	**0.195**	**0.286**	**0.196**	**0.396**	**0.409**	**0.296**
	rec_rate	**84.7%**	**62.1%**	84.5%	**45.1%**	**40.6%**	**63.4%**
(b) Results with $\overline{\mathcal{DB}}_{cmu}$ (MoCap from CMU dataset)							
Our App.	*rec_err*	0.222	0.337	0.243	0.429	0.561	0.358
	rec_rate	81.6%	50.6%	76.0%	37.6%	21.5%	53.5%
(c) Results with $\mathcal{DB}_{hdm}$ (MoCap from HDM05 dataset)							
Our App.	*rec_err*	0.317	0.406	0.237	0.554	0.595	0.422
	rec_rate	54.9%	29.3%	**85.4%**	6.4%	17.6%	38.7%

Table 6.5: Comparison with the state-of-the-art approaches on synthetically generated 2D image test dataset $\mathcal{TS}_1$. Both performance metrics the average reconstruction error *rec_err* and the reconstruction rate *rec_rate* are reported for all five action classes. **(a)** and **(b)** report results of the proposed approach on the databases $\mathcal{DB}_{cmu}$ and $\overline{\mathcal{DB}}_{cmu}$ respectively which are developed using CMU motion capture dataset, while **(c)** shows results when the database $\mathcal{DB}_{hdm}$ is used as the MoCap priors, which is based on HDM05 motion capture dataset.

better comparatively for all actions except for climbing action where the error increases a little bit due to the reason that after removing the actor's sequences, the MoCap dataset contains a very few examples of the climbing action, which ultimately results into increase in the reconstruction error. In contrast, if we consider the average results over all five actions, our approach executes a very good results as compared to the state-of-the-arts.

- For **third** case ($\mathcal{DB}_{hdm}$), when we use other MoCap dataset like HDM05 MoCap dataset. We observe that the reconstruction error increases and that is due to the skeleton discrepancies between CMU MoCap dataset and HDM05 MoCap dataset. For boxing action category, the reconstruction error is high because HDM05 MoCap dataset does not contain boxing poses at all. It just includes a few poses of punching and as a result the efficiency drops and the system executes high reconstruction error for that specific action. Even having different virtual marker placements and the skeleton discrepancies in HDM05 MoCap dataset, our approach still executes competitive results (see Table 6.5(c)).

- Our approach is more efficient with respect to run time as well and takes just 0.68 seconds per image for retrieval and reconstruction with feature set $\mathcal{F}^{\mathrm{im}}_{11}$ as reported in Table 6.4, while the state-of-the-art approach [RKS12] takes 5 seconds per image to converge.

Comparison with State-of-the-arts on Noisy Data						
Methods	**Err. Metrics**	***std*(0.0)**	***std*(0.1)**	***std*(0.2)**	***std*(0.3)**	***std*(0.4)**
[FZZW14]	*rec_err*	0.414	0.449	0.485	0.561	0.630
	rec_rate	32.6%	28.7%	24.4%	18.1%	13.1%
[RKS12]	*rec_err*	0.466	0.497	0.558	0.634	0.704
	rec_rate	23.9%	20.5%	13.8%	9.3%	4.8%
Our App.	*rec_err*	0.282	0.333	0.435	0.529	0.623
	rec_rate	66.2%	52.1%	37.7%	34.7%	33.1%

Table 6.6: Influence of the noise on overall reconstruction results. Both error categories, the average reconstruction error *rec_err* and the reconstruction rate *rec_rate*, are reported when Gaussian white noise with different standard deviations ***(std)*** is added into the 2D input query.

Noisy Input Data. In real world scenario, the 2D pose estimation from the real images are often ambiguous and noisy. To check the developed system's resistance against the noisy inputs, we test our 3D reconstruction approach on different levels of Gaussian white noise with standard deviation ***std***, starting from 0.0 (indicates no noise) to 0.1, 0.2, 0.3 and 0.4. Similar to [FZZW14], we also normalize the Gaussian white noise before adding it to the 2D synthetic test images. From the results reported in Table 6.6, we have found that our approach is more resistant to noise as compared to the state-of-the-arts. This is also evident from Figure 6.12, where our approach produces very good results even with the erroneous 2D joint positions. Moreover, the proposed system's resistance against noise can be evaluated when we give input the hand-drawn 2D sketches where the joint positions are very ambiguous and the 2D poses do not hold anthropometric regularity at all (see Figure 6.15).

6.6.5 Controlled Experiments

We analyze the influence of different parameters on our proposed approach by performing a few controlled experiments, which we discuss as follows.

Joints' Sensitivity. To see the joints' sensitivity with respect to the final reconstruction, we evaluate our approach joint-wise for all types of activity classes. We compute the average reconstruction error by computing the Euclidian distance between the estimated 3D poses and the ground truth poses for each individual joint. The results as illustrated in Figure 6.11 reveal that the joints like wrists, ankles and feet joints prove to be more erroneous and sensitive for all activities as compared to the other joints due to the reason that these joints have more capacity to move all around. On the other

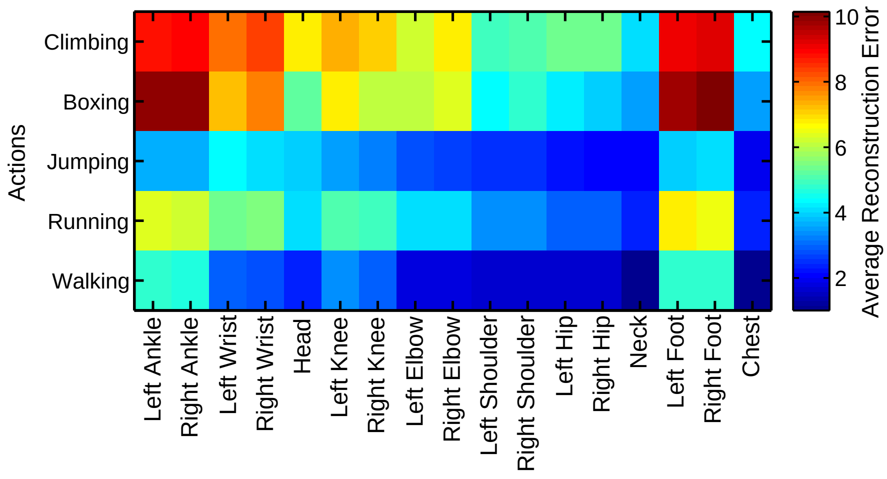


Figure 6.11: The sensitivity of each individual joint in all five action categories with reference to the average reconstruction error (Euclidean distance) which is color-coded.

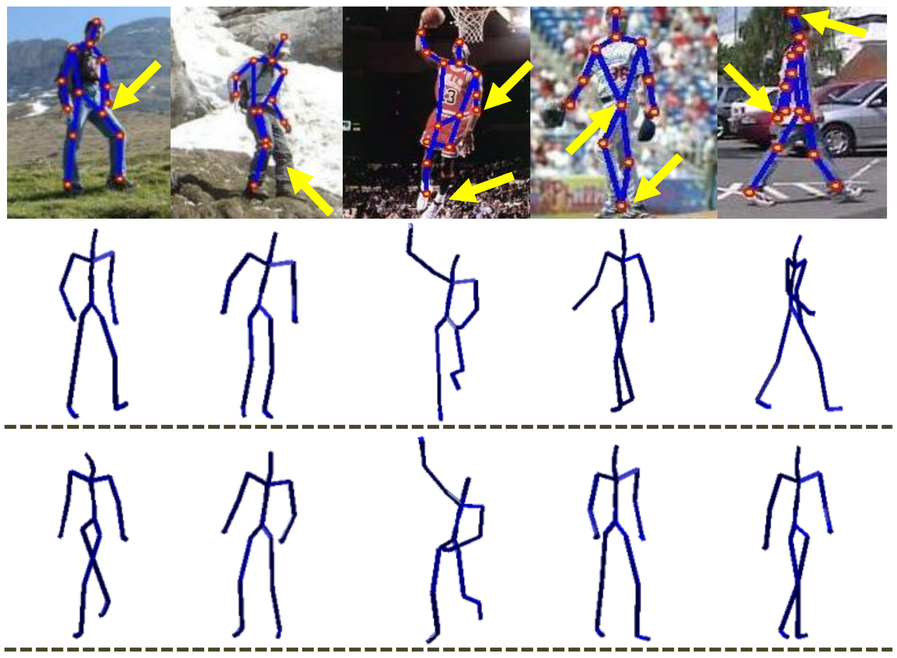


Figure 6.12: A few examples of outliers that are detected in the 2D input images when we employ off-the-shelf part-detector algorithm [YR11] to estimate the 2D pose, even then our reconstruction approach executes acceptable results. **First row** represents the input images while the **second** and **third rows** correspond to the estimated 3D poses at two arbitrary views.

hand, the joints like neck, shoulder and hip joints are considered to be less sensitive as expected due to limited movements.

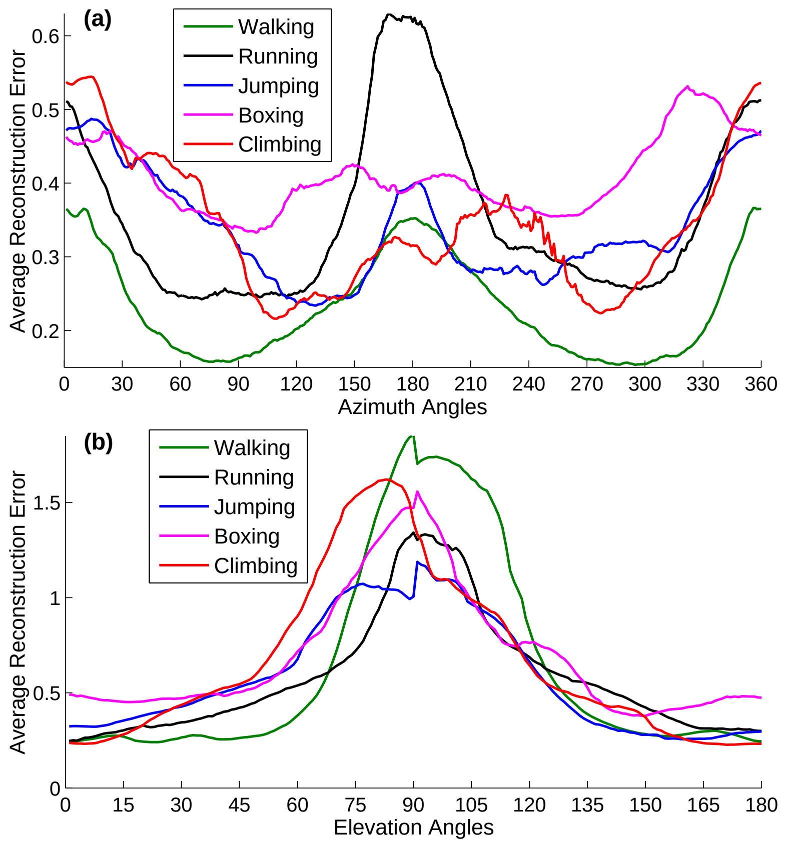


Figure 6.13: Influence of the test camera viewpoints in terms of average reconstruction error *rec_err* for all five actions when: **(a)** the elevation angle is fixed to 30 degree and the azimuth angles span from 0 to 360 degrees; **(b)** the azimuth angle is fixed to 30 degree and the elevation angles range from 0 to 180 degrees. We perform this experiment on 100 2D synthetic images for each action, which are selected randomly from the input test dataset $\mathcal{TS}_1$.

Camera Viewpoints. We also report on the influence of the test camera viewpoints on the reconstruction process with respect to azimuth and elevation angles. For that purpose, we perform evaluation on all possible camera viewpoints to check the robustness of the presented system. We perform this experiment on randomly selected 100 2D synthetic images for each action from the input test dataset $\mathcal{TS}_1$.

- In case of **azimuth angles**, we create the synthetic images for all five actions by weak perspective camera at the azimuth angles spanning from 0 to 360 degrees and fix the elevation angle to 30 degree. The results are presented in Figure 6.13(a), where we observe that our approach executes more error for the profile views than the frontal views for almost all action classes, but still produces acceptable results.
- In case of **elevation angles**, similar to first experiment, we generate the synthetic images for all five actions at the elevation angles ranges from 0 to 180 degrees when the azimuth angle is set to 30 degree. The results in Figure 6.13(b) reveal that

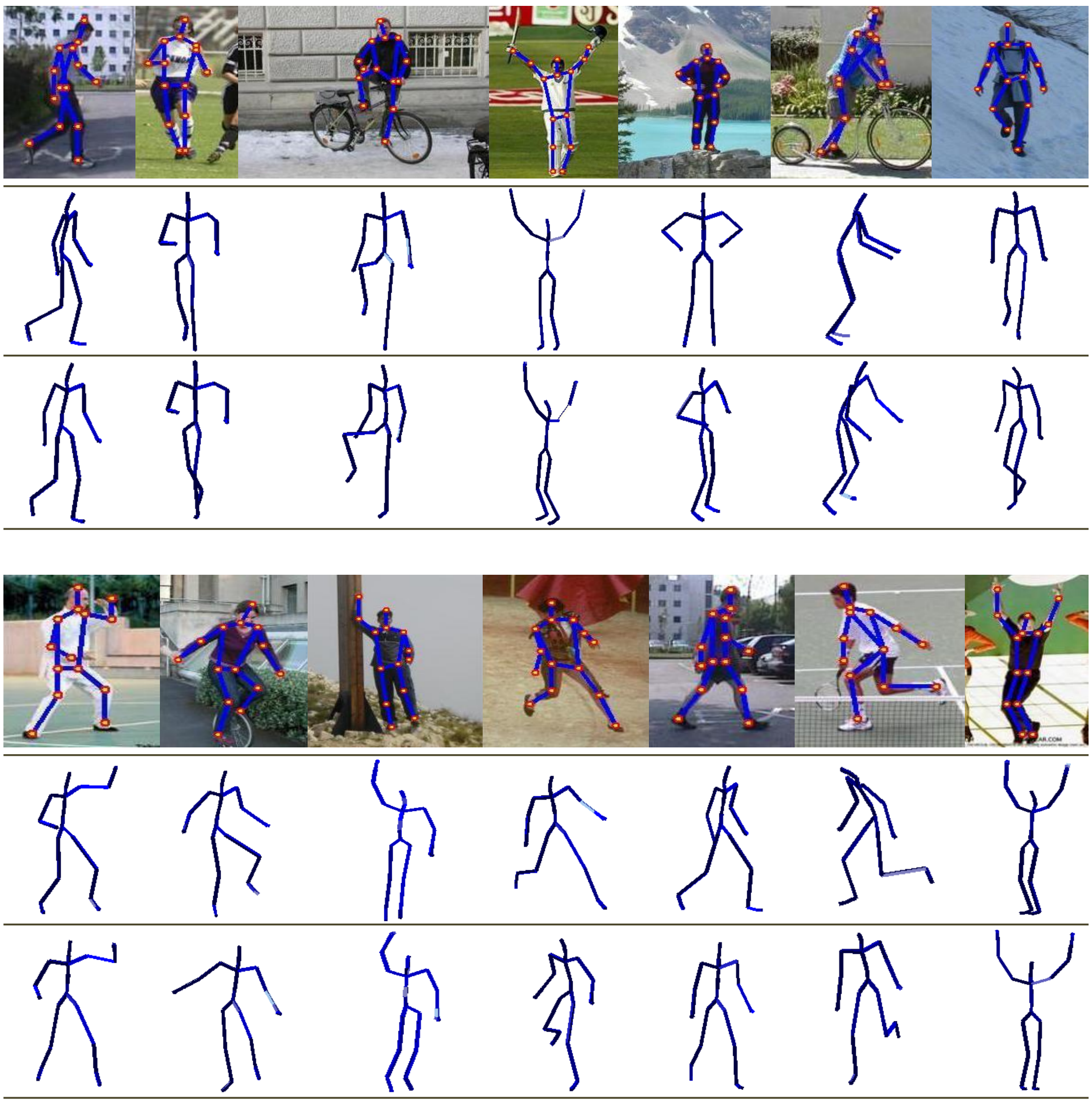


Figure 6.14: Qualitative evaluation on real images: A few examples of the reconstruction results on PARSE dataset [Ram07], when the automatic off-the-shelf part-detector algorithm [YR11] is utilized to estimate the 2D joint locations. **First rows** represent the real images with 2D estimated joint positions which are given as input, while the **second** and **third rows** are the corresponding 3D reconstructions at two different arbitrary viewpoints.

the presented system produces more reconstruction errors for the head-mounted camera views comparatively. At that view, the 2D input joint locations are overlapping with each other and become indistinctive which leads to the retrieval of inappropriate nearest neighbors and ultimately results into higher reconstruction error.

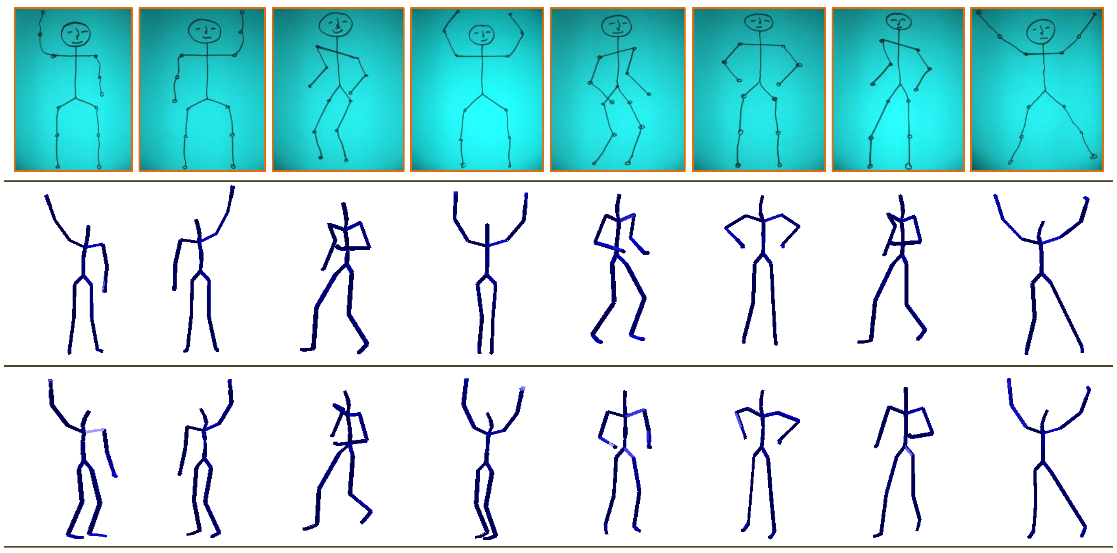


Figure 6.15: Qualitative evaluation on the hand-drawn sketches: A few examples of reconstruction on the hand-drawn sketches, when the 2D joint locations are manually labelled. **First row** represents the hand-drawn sketches that are given as input, while the **second** and **third rows** are the corresponding 3D reconstructions at two different arbitrary viewpoints.

6.6.6 Qualitative Evaluation

6.6.6.1 Real Images

We employ PARSE dataset [Ram07] for qualitative evaluation of our approach on the real images. We use the automatic off-the-shelf part-detector algorithm [YR11] in order to estimate the 2D joint locations in contrast to the state-of-the-art approaches [FZZW14, RKS12], where the authors manually annotate the 2D joint positions. The off-the-shelf part-detector algorithm produces more noisy 2D joint locations comparatively. Some qualitative reconstruction results based on the 2D poses estimated from the real images are reported in Figure 6.14. Although the 2D estimated joint positions are noisy and ambiguous, our approach executes plausible and robust 3D reconstruction results. A few more examples are shown in Figure 6.12, where 2D input poses are invalid and ambiguous, even having such a erroneous input our approach produces good results.

6.6.6.2 Hand-drawn Sketches

We also evaluate our proposed methodology qualitatively on the hand-drawn sketches, which we draw for different action categories like walking, jumping jack and grabbing action. The inference of the 3D poses from the hand-drawn 2D sketches is the most challenging scenario due to the reason that: the non-existence anthropometric property of the hand-drawn poses; the varying lengths and sizes of different body segments; the unnatural body part movements *etc.* We first manually label the 2D joint positions of the given 2D hand-drawn sketch. Although giving such a noisy and ambiguous input, our system still produces plausible 3D poses. A few qualitative reconstruction results on the basis of hand-drawn sketches are reported in Figure 6.15.

6.7 Conclusion

We have presented in this chapter an efficient and robust framework for the 3D pose retrieval leading towards the 3D pose reconstruction from a single 2D image either synthetic, real or hand-drawn. For 3D pose retrieval from the MoCap dataset, we develop a set of feature sets based on different combinations of joints. We evaluate the developed feature sets in terms of similarity measure, reconstruction error, time and memory consumption. We introduce the two-fold method to estimate the camera parameters through sampling of the MoCap data, the retrieved Knn and the projective constraints. We also exploit the retrieved 3D similar poses as pose priors and derive benefits form the joint weights in proposed reconstruction approach in order to infer the final 3D pose. We perform quantitative analysis on 43809 synthetic images and qualitative analysis on the internet real images as well as on the hand-drawn sketches. On this large input testing dataset and with a variety of experimental setups based on different MoCap datasets and inputs, we have evaluated our proposed framework and found that our approach outperforms all existing state-of-the-art approaches even in case of noisy input images. Our retrieval and reconstruction approach takes roughly 2 poses per second.

7

A Dual-Source Approach for 3D Pose Estimation from a Single Image

True wisdom comes to each of us when we realize how little we understand about life, ourselves, and the world around us.

Socrates

7.1 Introduction

Human 3D pose estimation from a single RGB image is a very challenging task. One approach to solve this task is to collect training data, where each image is annotated with the 3D pose. A regression model, for instance, can then be learned to predict the 3D pose from the image [BS10, KG14, ICS14, AT06b, BSKM08]. In contrast to 2D pose estimation, however, acquiring accurate 3D poses for an image is very elaborate. Popular datasets like HumanEva [SBB10] or Human3.6M [IPOS14] synchronized cameras with a commercial marker-based system to obtain 3D poses for images. This requires a very expensive hardware setup and the requirements for the marker-based system like controlled indoor environment and the markers attached with performing actor prevent the capturing of realistic natural images.

Instead of training a model on pairs consisting of an image and a 3D pose, we propose an approach that is able to incorporate 2D and 3D information from two different training sources. The first source consists of images with annotated 2D pose. Since 2D poses in images can be manually annotated, they do not impose any constraints regarding the environment from where the images are taken. Indeed any image from the Internet can be taken and annotated. The second source is accurate 3D motion capture data captured in a lab, *e.g.*, as in the CMU motion capture dataset [CMU14] or the Human3.6M dataset [IPOS14]. We consider both sources as independent, *i.e.*, we do not know the 3D pose for any image. To integrate both sources, we propose a dual-source approach as illustrated in Figure 7.1. To this end, we first convert the

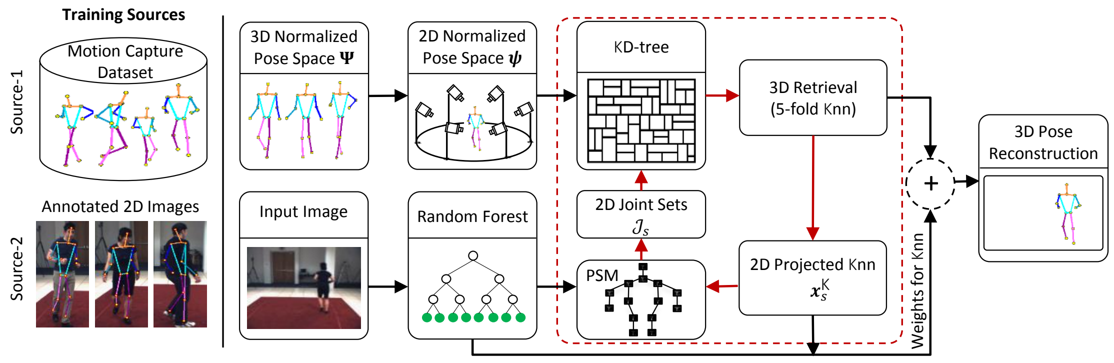


Figure 7.1: Overview. Our approach relies on two training sources. The first source is a motion capture database that contains only 3D poses. The second source is an image database with annotated 2D poses. The motion capture data is processed by pose normalization and projecting the poses to 2D using several virtual cameras. This gives many 3D-2D pairs where the 2D poses serve as features. The image data is used to learn a pictorial structure model (PSM) for 2D pose estimation where the unaries are learned by a random forest. Given a test image, the PSM predicts the 2D pose which is then used to retrieve the normalized nearest 3D poses. The final 3D pose is then estimated by minimizing the projection error under the constraint that the solution is close to the retrieved poses, which are weighted by the unaries of the PSM. The steps (*red arrows*) in the dashed box can be iterated by updating the binaries of the PSM using the retrieved poses and updating the 2D pose.

motion capture data into a normalized 2D pose space and learn a regressor for 2D pose estimation from the image data. During inference, we first estimate 2D pose and create Knn search to retrieve the nearest 3D poses using an approach that is robust to 2D pose estimation errors. We then jointly estimate a mapping from the 3D pose space to the image, weight the retrieved nearest poses according to the image evidence, identify wrongly estimated 2D joints, and estimate the 3D pose. During this process, the 2D pose can also be refined and the approach can be iterated to update the estimated 3D and 2D pose.

We evaluate our approach on two popular datasets for 3D pose estimation. On both datasets, our approach achieves state-of-the-art results when using both sources from the same dataset, but it even achieves competitive results when the motion capture data is taken from a very different dataset. We provide a thorough evaluation of the proposed approach. In particular, we analyze the impact of differences of the skeleton structure between the two training sources, the impact of the accuracy of the used 2D pose estimator, and the impact of the similarity of the training and test poses.

7.2 Overview

In this work, we aim to predict the 3D pose from an RGB image. Since acquiring 3D pose data in natural environments is impractical and annotating 2D images with 3D pose data is infeasible, we do not assume that our training data consists of images annotated with 3D pose. Instead, we propose an approach that utilizes two independent sources of training data. The first source consists of motion capture data, which is publically available in large quantities and that can be recorded in controlled environments. The second source consists of images with annotated 2D poses, which is also available and can be easily provided by humans. Since we do not assume that we know any relations between the sources except that the motion capture data includes the poses we are interested in, we preprocess the sources first independently as illustrated in Figure 7.1. From the image data, we learn a pictorial structure model (PSM) to predict 2D poses from images. This will be discussed in Section 7.3. The motion capture data is prepared to efficiently retrieve 3D poses that could correspond to a 2D pose. This part is described in Section 7.4.1. We will show that the retrieved poses are insufficient for estimating the 3D pose. Instead, we estimate the pose by minimizing the projection error under the constraint that the solution is close to the retrieved poses (Section 7.4.2). In addition, the retrieved poses can be used to update the PSM and the process can be iterated (Section 7.4.3). In our experiments, we show that we achieve very good results for 3D pose estimation with only one or two iterations.

7.3 2D Pose Estimation

In this work, we adopt a PSM that represents the 2D body pose $\mathbf{x}$ with a graph $\mathcal{G} = (\mathcal{J}, \mathcal{L})$, where each vertex corresponds to 2D coordinates of a particular body joint i, and edges correspond to the kinematic constraints between two joints i and j. We assume that the graph is a tree structure which allows efficient inference. Given an image $\mathbf{I}$, the 2D body pose is inferred by maximizing the following posterior distribution,

$$P(\mathbf{x}|\mathbf{I}) \propto \prod_{i \in \mathcal{J}} \phi_i(x_i|\mathbf{I}) \prod_{(i,j) \in \mathcal{L}} \phi_{i,j}(x_i, x_j), \tag{7.1}$$

where the unary potentials $\phi_i(x_i|\mathbf{I})$ correspond to joint templates and define the probability of the i^{th} joint at location x_i. The binary potentials $\phi_{i,j}(x_i, x_j)$ define the deformation cost of joint i from its parent joint j in the tree structure.

The unary potentials in (7.1) can be modeled by any discriminative model, *e.g.*, SVM in [YR11] or random forests in [DLGVG14]. In this work, we choose random

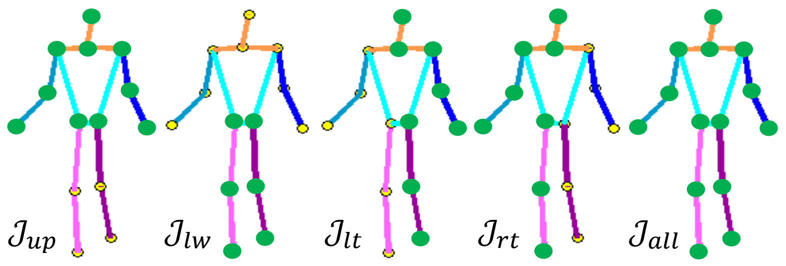


Figure 7.2: Different joint sets. $\mathcal{J}_{up}$ is based on upper body joints, $\mathcal{J}_{lw}$ lower body joints, $\mathcal{J}_{lt}$ left body joints, $\mathcal{J}_{rt}$ right body joints and $\mathcal{J}_{all}$ is composed of all body joints. The selected joints are indicated by the large *green* circles.

forest based joint regressors. We train a separate joint regressor for each body joint. Following [DLGVG14], we model binary potentials for each joint i as a Gaussian mixture model with respect to its parent j. We obtain the relative joint offsets between two adjacent joints in the tree structure and cluster them into $c = 1, \ldots, C$ clusters using k-means clustering. The offsets in each cluster are then modeled with a weighted Gaussian distribution as,

$$\phi_{ij}(x_i, x_j) = w_{ij}^c \exp\left(-\frac{1}{2}\left(d_{ij} - \mu_{ij}^c\right)^T \left(\Sigma_{ij}^c\right)^{-1} \left(d_{ij} - \mu_{ij}^c\right)\right) \tag{7.2}$$

with mean μ_{ij}^c, covariance Σ_{ij}^c and $d_{ij} = (x_i - x_j)$. The weights w_{ij}^c are set according to the cluster frequency $P(c|i,j)^\alpha$ with a normalization constant $\alpha = 0.1$ [DLGVG14].

7.4 3D Pose Estimation

While the PSM for 2D pose estimation is trained on the images with 2D pose annotations as shown in Figure 7.1, we now describe an approach that makes use of a second dataset with 3D poses in order to predict the 3D pose from an image. Since the two sources are independent, we first have to establish relations between 2D poses and 3D poses. This is achieved by using an estimated 2D pose as query for 3D pose retrieval (Section 7.4.1). The retrieved poses, however, contain many wrong poses due to errors in 2D pose estimation, 2D-3D ambiguities and differences of the skeletons in the two training sources. It is therefore necessary to fit the 3D poses to the 2D observations. This will be described in Section 7.4.2.

7.4.1 3D Pose Retrieval

In order to efficiently retrieve 3D poses for a 2D pose query, we preprocess the motion capture data. We first normalize the poses by discarding orientation and translation information from the poses in our motion capture database. We denote a 3D normalized pose with $\mathbf{X}$ and the 3D normalized pose space with $\boldsymbol{\Psi}$. As in [YKW13], we project the normalized poses $\mathbf{X} \in \boldsymbol{\Psi}$ to 2D using orthographic projection. We use 144 virtual camera views with azimuth angles spanning 360 degrees and elevation angles in the range of 0 and 90 degree. Both angles are uniformly sampled with step size of 15 degree. We further normalize the projected 2D poses by scaling them such that the y-coordinates of the joints are within the range of $[-1, 1]$. The normalized 2D pose space is denoted by ψ and does not depend on a specific camera model or coordinate system. This step is illustrated in Figure 7.1. After a 2D pose is estimated by the approach described in Section 7.3, we first normalize it according to ψ, *i.e.*, we translate and scale the pose such that the y-coordinates of the joints are within the range of $[-1, 1]$, then use it to retrieve 3D poses. The distance between two normalized 2D poses is given by the average Euclidean distance of the joint positions. The K-nearest neighbors in ψ are efficiently retrieved by a kd-tree [KTWZ10]. The retrieved normalized 3D poses are the corresponding poses in $\boldsymbol{\Psi}$. An incorrect 2D pose estimation or even an imprecise estimation of a single joint position, however, can effect the accuracy of the 3D pose retrieval and consequently the 3D pose estimation. We therefore propose to use several 2D joint sets for pose retrieval where each joint set contains a different subset of all joints. The joint sets are shown in Figure 7.2. While $\mathcal{J}_{all}$ contains all joints, the other sets $\mathcal{J}_{up}$, $\mathcal{J}_{lw}$, $\mathcal{J}_{lt}$ and $\mathcal{J}_{rt}$ contain only the joints of the upper body, lower body, left hand side and right hand side of the body, respectively. In this way we are able to compensate for 2D pose estimation errors, if at least one of our joint sets does not depend on the wrongly estimated 2D joints.

7.4.2 3D Pose Estimation

In order to obtain the 3D pose $\mathbf{X}$, we have to estimate the unknown projection $\mathcal{M}$ from the normalized pose space $\boldsymbol{\Psi}$ to the image and infer which joint set $\mathcal{J}_s$ explains the image data best. To this end, we minimize the energy

$$E(\mathbf{X}, \mathcal{M}, s) = \omega_p E_p(\mathbf{X}, \mathcal{M}, s) + \omega_r E_r(\mathbf{X}, s) + \omega_a E_a(\mathbf{X}, s) \tag{7.3}$$

consisting of the three weighted terms E_p, E_r and E_a.

The first term $E_p(\mathbf{X}, \mathcal{M}, s)$ measures the projection error of the 3D pose $\mathbf{X}$ and the projection $\mathcal{M}$:

$$E_p(\mathbf{X}, \mathcal{M}, s) = \left(\sum_{i \in \mathcal{J}_s} \|\mathcal{M}(X_i) - x_i\|^2 \right)^{\frac{1}{4}}, \tag{7.4}$$

where x_i is the joint position of the predicted 2D pose and X_i is the 3D joint position of the unknown 3D pose. The parameter s defines the set of valid 2D joint estimates and the error is only computed for the joints of the corresponding joint set $\mathcal{J}_s$, *e.g.*, only the joints of the upper body are used for $\mathcal{J}_{up}$.

The second term enforces that the pose $\mathbf{X}$ is close to the retrieved 3D poses $\mathbf{X}_s^{\mathsf{k}}$ for a joint set $\mathcal{J}_s$:

$$E_r(\mathbf{X}, s) = \sum_{\mathsf{k}} w_{\mathsf{k},s} \left(\sum_{i \in \mathcal{J}_{all}} \| X_{s,i}^{\mathsf{k}} - X_i \|^2 \right)^{\frac{1}{4}} . \tag{7.5}$$

In contrast to (7.4), the error is computed over all joints but the set of nearest neighbors depends on s. In our experiments, we will show that an additional weighting of the nearest neighbors by $w_{\mathsf{k},s}$ improves the 3D pose estimation accuracy.

Although the term $E_r(\mathbf{X}, s)$ penalizes already deviations from the retrieved poses and therefore enforces implicitly anthropometric constraints, we found it useful to add an additional term that enforces anthropometric constraints on the limbs:

$$E_a(\mathbf{X}, s) = \sum_{\mathsf{k}} w_{\mathsf{k},s} \left(\sum_{(i,j) \in \mathcal{L}} \left(L_{s,i,j}^{\mathsf{k}} - L_{i,j} \right)^2 \right)^{\frac{1}{4}} , \tag{7.6}$$

where $L_{i,j}$ denotes the limb length between two joints.

Minimizing the energy $E(\mathbf{X}, \mathcal{M}, s)$ (7.3) over the discrete variable s and the continuous parameters $\mathbf{X}$ and $\mathcal{M}$ would be expensive. We therefore propose to obtain an approximate solution where we estimate the projection $\mathcal{M}$ first. For the projection, we assume that the extrinsic parameters are given and only estimate the global orientation and translation. The projection $\hat{\mathcal{M}}_s$ is estimated for each joint set $\mathcal{J}_s$ with $s \in \{up, lw, lt, rt, all\}$ by minimizing

$$\hat{\mathcal{M}}_s = \operatorname*{arg\,min}_{\mathcal{M}} \left\{ \sum_{\mathsf{k}=1}^{\mathsf{K}} E_p(\mathbf{X}_s^{\mathsf{k}}, \mathcal{M}, s) \right\} \tag{7.7}$$

using non-linear gradient optimization. Given the estimated projections $\hat{\mathcal{M}}_s$ for each joint set, we then optimize over the discrete variable s:

$$\hat{s} = \operatorname*{arg\,min}_{s \in \{up, lw, lt, rt, all\}} \left\{ \sum_{\mathsf{k}=1}^{\mathsf{K}} E(\mathbf{X}_s^{\mathsf{k}}, \hat{\mathcal{M}}_s, s) \right\} . \tag{7.8}$$

As a result, we obtain $\hat{s}$ and $\hat{\mathcal{M}} = \hat{\mathcal{M}}_{\hat{s}}$ and finally minimize

$$\widehat{\mathbf{X}} = \operatorname*{arg\,min}_{\mathbf{X}} \left\{ E(\mathbf{X}, \hat{\mathcal{M}}, \hat{s}) \right\} \tag{7.9}$$

to obtain the 3D pose.

Implementation details. Instead of obtaining $\hat{s}$ by minimizing (7.8), $\hat{s}$ can also be estimated by maximizing the posterior distribution for the 2D pose (7.1). To this end, we project all retrieved 3D poses to the image by

$$x^{\mathsf{k}}_{s,i} = \hat{\mathcal{M}}_s \left(X^{\mathsf{k}}_{s,i} \right). \tag{7.10}$$

Note that the retrieved poses contain all joints although only a subset of joints was used for retrieval. The binary potentials $\phi_{i,j}(x_i, x_j|\mathbf{X}_s)$, which are mixture of Gaussians, are then computed from the projected full body poses for each set,

$$P(\mathbf{x}|\mathbf{X}_s, \mathbf{I}) \propto \prod_{i \in \mathcal{J}} \phi_i(x_i|\mathbf{I}) \prod_{i,j \in \mathcal{L}} \phi_{i,j}(x_i, x_j|\mathbf{X}_s), \tag{7.11}$$

and $\hat{s}$ is inferred by the maximum posterior probability:

$$(\hat{\mathbf{x}}, \hat{s}) = \arg\max_{\mathbf{x},s} \left\{ P(\mathbf{x}|\mathbf{X}_s, \mathbf{I}) \right\}, \tag{7.12}$$

which can be efficiently computed since the terms $\phi_i(x_i|\mathbf{I})$ have been already computed for 2D pose estimation. Besides of the joint set, we also obtain a refined 2D pose $\hat{\mathbf{x}}$, which is used finally to compute the projection error $E_p(\mathbf{X}, \hat{\mathcal{M}}, \hat{s})$ in (7.9).

For 3D pose estimation, we only keep the retrieved poses from the inferred joint set $\hat{s}$ and weight each pose by the unaries (7.1)

$$\mathsf{w}_{\mathsf{k},s} = \sum_{i \in \mathcal{J}} \phi_i(x^{\mathsf{k}}_{s,i}|\mathbf{I}), \tag{7.13}$$

and normalized by

$$w_{\mathsf{k},s} = \frac{\mathsf{w}_{\mathsf{k},s} - \min_{\mathsf{k}'}(\mathsf{w}_{\mathsf{k}',s})}{\max_{\mathsf{k}'}(\mathsf{w}_{\mathsf{k}',s}) - \min_{\mathsf{k}'}(\mathsf{w}_{\mathsf{k}',s})}. \tag{7.14}$$

For the retrieved poses, we only keep the K_w poses with the highest weights. In our experiments, we show that good results are achieved with $\mathsf{K} = 256$ and $\mathsf{K}_w = 64$. The dimensionality of $\mathbf{X}$ can be reduced by applying PCA to the weighted poses. We thoroughly evaluate the impact of the implementation details in Section 7.5.1.1.

7.4.3 Iterative Approach

The approach can be iterated by using the refined 2D pose $\hat{\mathbf{x}}$ (7.12) as query for 3D pose retrieval (Section 7.4.1) as illustrated in Figure 7.1. Having more than one iteration is not very expensive since many terms like the unaries need to be computed only once and the optimization of (7.7) can be initialized by the results of the previous iteration. The final pose estimation described in Section 7.4.2 also needs to be computed only once after the last iteration. In our experiments, we show that two iterations are sufficient.

7.5 Experiments

We evaluate the proposed approach on two publicly available datasets, namely HumanEva-I [SBB10] and Human3.6M [IPOS14]. Both datasets provide accurate 3D poses for each image and camera parameters. For both datasets, we use a skeleton consisting of 14 joints, namely head, neck, ankles, knees, hips, wrists, elbows and shoulders. For evaluation, we use the *3D pose error* as defined in [SSRA+12]. The error measures the accuracy of the relative pose up to a rigid transformation. To this end, the estimated skeleton is aligned to the ground-truth skeleton by a rigid transformation and the average 3D Euclidean joint error after alignment is measured. In addition, we use the CMU motion capture dataset [CMU14] as training source.

7.5.1 Evaluation on HumanEva-I Dataset

We follow the same protocol as described in [SSQTMN13, KG14] and use the provided training data to train our approach while using the validation data as test set. As in [SSQTMN13, KG14], we report our results on every 5th frame of the sequences *walking* (A1) and *jogging* (A2) for all three subjects (S1, S2, S3) and camera C1. For 2D pose estimation, we train regression forests and PSMs for each activity as described in [DLGVG14]. The regression forests for each joint consists of 8 trees, each trained on 700 randomly selected training images from a particular activity. While we use $c = 15$ mixtures per part (7.2) for the initial pose estimation, we found that 5 mixtures are enough for pose refinement (Section 7.4.2) since the retrieved 2D nearest neighbors strongly reduce the variation compared to the entire training data. In our experiments, we consider two sources for the motion capture data, namely HumanEva-I and the CMU motion capture dataset. We first evaluate the parameters of our approach using the entire 49K 3D poses of the HumanEva training set as motion capture data. Although the training data for 2D pose estimation and the 3D pose data are from the same dataset, the sources are separated and it is unknown which 3D pose corresponds to which image.

7.5.1.1 Parameters

Joint Sets $\mathcal{J}$. For 3D pose retrieval (Section 7.4.1), we use several joint sets $\mathcal{J}_s$ with $s \in \{up, lw, lt, rt, all\}$. For the evaluation, we use only one iteration and $\mathsf{K} = 256$

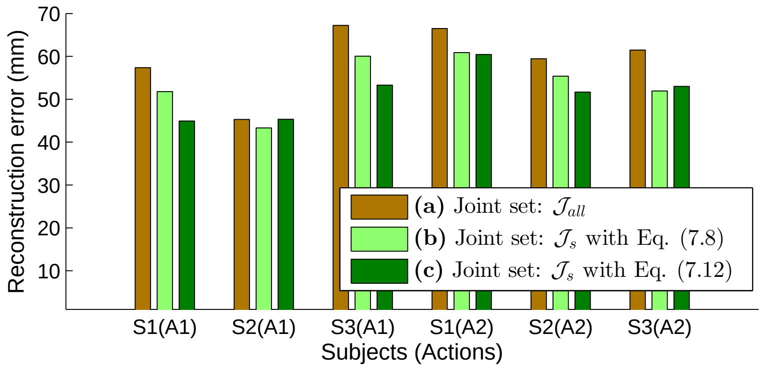


Figure 7.3: **(a)** Using only joint set $\mathcal{J}_{all}$. **(b)** Using all joint sets $\mathcal{J}_s$ and estimating $\hat{s}$ using (7.8). **(c)** All joint sets $\mathcal{J}_s$ and estimating $\hat{s}$ using (7.12).

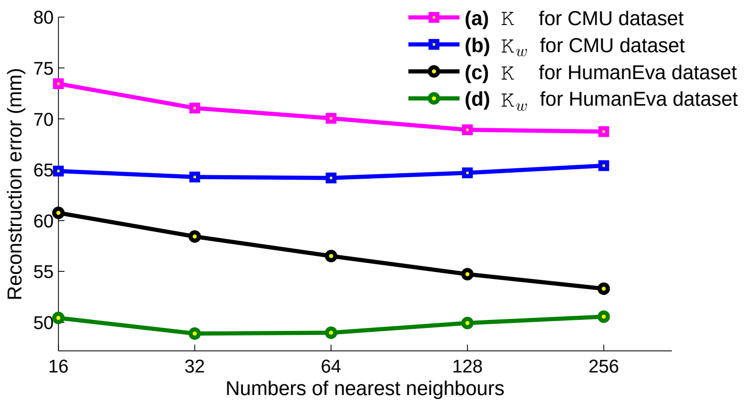


Figure 7.4: Impact of number of nearest neighbors K and weighting of nearest neighbors K_w. The results are reported for subject S3 with *walking* action (A1, C1) using the CMU dataset **(a-b)** and HumanEva **(c-d)** for 3D pose retrieval.

without weighting. The results in Figure 7.3 show the benefit of using several joint sets. Estimating $\hat{s}$ using (7.12) instead of (7.8) also reduces the pose estimation error.

Nearest Neighbors K_w. Our 3D pose estimation (Section 7.4.2) depends on the retrieved K 3D poses, which are then weighted and reduced to K_w. The impact of the weighting and the number of nearest neighbors is evaluated in Figure 7.4. The results show that the weighting reduces the pose estimation error independently of the used motion capture dataset. Without weighting more nearest neighbors are required. If not otherwise specified, we use $K = 256$ and $K_w = 64$ for the rest of the paper. We also evaluated our approach without a 3D pose model. In this case, we take the average pose of the retrieved K or K_w poses. If the average of the retrieved K or K_w poses is used instead of optimizing (7.9), the errors are $55.7mm$ and $48.9mm$, respectively, whereas $53.2mm$ and $47.5mm$ by optimizing (7.9). PCA can be used to reduce the dimension

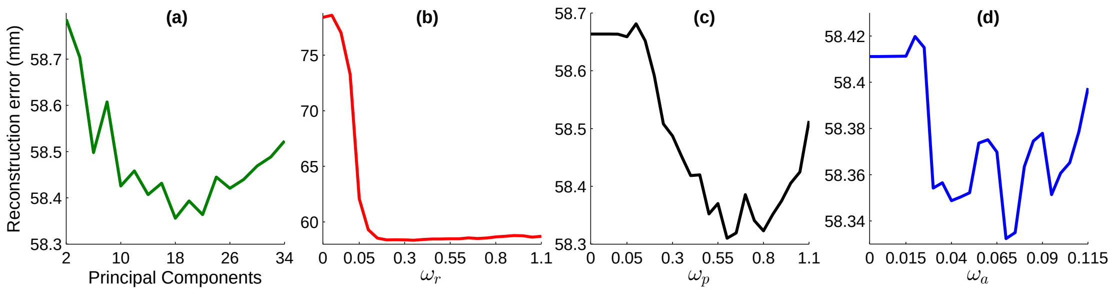


Figure 7.5: (a) Impact of number of eigenposes. The error is reported for subject S3 with action *jogging* (A2, C1) using the CMU dataset for 3D pose retrieval. **(b-d)** Impact of weights ω_r, ω_p and ω_a in (7.3).

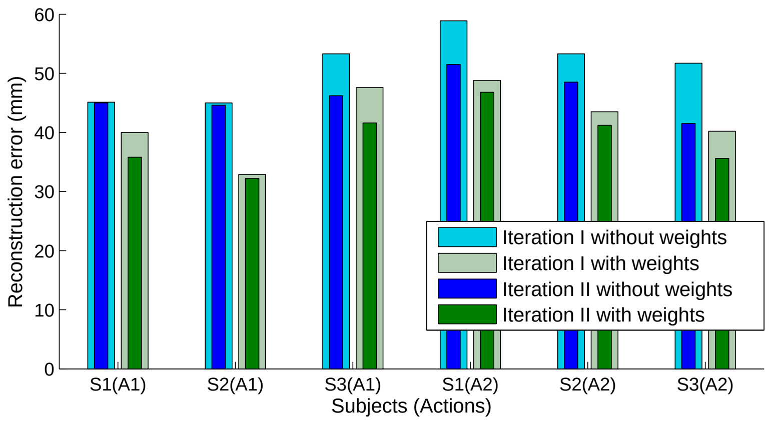


Figure 7.6: Impact of number of iterations and weighting of nearest neighbors.

of **X**. Figure 7.5(a) evaluates the impact of the number of principal components. Good results are achieved for 10-26 components, but the exact number is not critical. In our experiments, we use 18.

Energy Terms. In order to recover 3D pose (Section 7.4.2), we use an energy (7.3) consisting of three weighted terms, namely E_r, E_p and E_a. The impact of the weights is reported in Figure 7.5(b-d). Without the term E_r, the error is very high. This is expected since the projection error E_p is evaluated on the joint set $\mathcal{J}_{\hat{s}}$. If $\mathcal{J}_{\hat{s}}$ does not contain all joints, the optimization is not sufficiently constrained without E_r. Since E_r is already weighted by the image consistency of the retrieved poses, E_p does not result in a large drop of the error, but refines the 3D pose. The additional anthropometric constraints E_a slightly reduce the error in addition. In our experiments, we use $\omega_p = 0.55$, $\omega_r = 0.35$, and $\omega_a = 0.065$.

(a) Walking Sequences (A1, C1) of HumanEva-I dataset				
Methods	**Walking (A1, C1)**			**Average**
	S1	**S2**	**S3**	
Kostrikov *et al.* [KG14]	44.0 ± 15.9	30.9 ± 12.0	41.7 ± 14.9	38.9 ± 14.3
Wang *et al.* [WWL+14]	71.9 ± 19.0	75.7 ± 15.9	85.3 ± 10.3	77.6 ± 15.1
Simo-Serra *et al.* [SSQTMN13]	65.1 ± 17.4	48.6 ± 29.0	73.5 ± 21.4	62.4 ± 22.6
Radwan *et al.* [RDG13]	75.1 ± 35.6	99.8 ± 32.6	93.8 ± 19.3	89.6 ± 29.2
Simo-Serra *et al.* [SSRA+12]	99.6 ± 42.6	108.3 ± 42.3	127.4 ± 24.0	111.8 ± 36.3
Bo *et al.* [BS10] (GT-BB)	46.4 ± 20.3	**30.3** ± 10.5	64.9 ± 35.8	47.2 ± 22.2
Bo *et al.* [BS10] (Est-BB)	54.8 ± 40.7	36.7 ± 20.5	71.3 ± 39.8	54.3 ± 33.7
Bo *et al.* [BS10]*	38.2 ± 21.4	32.8 ± 23.1	40.2 ± 23.2	37.1 ± 22.6
(i) Our Approach (MoCap from HumanEva-I dataset)				
Iteration-I	40.1 ± 34.5	33.1 ± 27.7	47.5 ± 35.2	40.2 ± 32.5
Iteration-II	**35.8** ± 34.0	32.4 ± 26.9	**41.6** ± 35.4	**36.6** ± 32.1
(ii) Our Approach (MoCap from CMU dataset)				
Iteration-I	54.5 ± 23.7	54.2 ± 21.4	64.2 ± 26.7	57.6 ± 23.9
Iteration-II	52.2 ± 20.5	51.0 ± 15.1	62.8 ± 27.4	55.3 ± 21.0

(b) Jogging Sequences (A2, C1) of HumanEva-I dataset				
Methods	**Jogging (A2, C1)**			**Average**
	S1	**S2**	**S3**	
Kostrikov *et al.* [KG14]	57.2 ± 18.5	**35.0** ± 9.9	**33.3** ± 13.0	41.8 ± 13.8
Wang *et al.* [WWL+14]	62.6 ± 10.2	77.7 ± 12.1	54.4 ± 9.0	64.9 ± 10.4
Simo-Serra *et al.* [SSQTMN13]	74.2 ± 22.3	46.6 ± 24.7	32.2 ± 17.5	51.0 ± 21.5
Radwan *et al.* [RDG13]	79.2 ± 26.4	89.8 ± 34.2	99.4 ± 35.1	89.5 ± 31.9
Simo-Serra *et al.* [SSRA+12]	109.2 ± 41.5	93.1 ± 41.1	115.8 ± 40.6	106.0 ± 41.1
Bo *et al.* [BS10] (GT-BB)	64.5 ± 27.5	48.0 ± 17.0	38.2 ± 17.7	50.2 ± 20.7
Bo *et al.* [BS10] (Est-BB)	74.2 ± 47.1	51.3 ± 18.1	48.9 ± 34.2	58.1 ± 33.1
Bo *et al.* [BS10]*	42.0 ± 12.9	34.7 ± 16.6	46.4 ± 28.9	41.0 ± 19.5
(i) Our Approach (MoCap from HumanEva-I dataset)				
Iteration-I	48.6 ± 33.3	43.6 ± 31.5	40.0 ± 27.9	44.1 ± 30.9
Iteration-II	**46.6** ± 30.4	41.4 ± 31.4	35.4 ± 25.2	**41.1** ± 29.0
(ii) Our Approach (MoCap from CMU dataset)				
Iteration-I	76.2 ± 23.8	74.5 ± 19.6	58.3 ± 23.7	69.7 ± 22.4
Iteration-II	74.5 ± 23.2	72.4 ± 20.6	56.8 ± 21.4	67.9 ± 21.7

Table 7.1: Comparison with other state-of-the-art approaches on the HumanEva-I dataset for actions *walking* (A1, C1) represented in **(a)** and *jogging* (A2, C2) shown in **(b)**. The average 3D pose error (mm) and standard deviation are reported for all three subjects (S1, S2, S3) and camera C1. * denotes a different evaluation protocol. **(i)** Results of the proposed approach with one or two iterations and motion capture data from the HumanEva-I dataset. **(ii)** Results with motion capture data from the CMU dataset.

Results Summary with Average Results on HumanEva-I dataset			
Methods	**Average Results**		**Total Average**
	Walking (A1, C1)	**Jogging (A2, C1)**	
Kostrikov *et al.* [KG14]	38.9 ± 14.3	41.8 ± 13.8	40.4 ± 14.1
Wang *et al.* [WWL+14]	77.6 ± 15.1	64.9 ± 10.4	71.3 ± 12.8
Simo-Serra *et al.* [SSQTMN13]	62.4 ± 22.6	51.0 ± 21.5	56.7 ± 22.1
Radwan *et al.* [RDG13]	89.6 ± 29.2	89.5 ± 31.9	89.5 ± 30.5
Simo-Serra *et al.* [SSRA+12]	111.8 ± 36.3	106.0 ± 41.1	108.9 ± 38.7
Bo *et al.* [BS10] (GT-BB)	47.2 ± 22.2	50.2 ± 20.7	48.7 ± 21.5
Bo *et al.* [BS10] (Est-BB)	54.3 ± 33.7	58.1 ± 33.1	56.2 ± 33.4
Bo *et al.* [BS10]*	37.1 ± 22.6	41.0 ± 19.5	39.1 ± 21.1
(i) Our Approach (MoCap from HumanEva-I dataset)			
Iteration-I	40.2 ± 32.5	44.1 ± 30.9	42.2 ± 31.7
Iteration-II	**36.6** ± 32.1	**41.1** ± 29.0	**38.9** ± 30.6
(ii) Our Approach (MoCap from CMU dataset)			
Iteration-I	57.6 ± 23.9	69.7 ± 22.4	63.7 ± 23.2
Iteration-II	55.3 ± 21.0	67.9 ± 21.7	61.6 ± 21.4

Table 7.2: Comparison with other state-of-the-art approaches on average results for all three subjects (S1, S2, S3) with actions *walking* (A1, C1) and *jogging* (A2, C2). * denotes a different evaluation protocol. **(i)** Results of the proposed approach with one or two iterations and motion capture data from the HumanEva-I dataset. **(ii)** Results with motion capture data from the CMU dataset.

Iterations. We finally evaluate the benefit of having more than one iteration (Section 7.4.3). Figure 7.6 compares the pose estimation error for one and two iterations. For completeness, the results for nearest neighbors without weighting are included. In both cases, a second iteration decreases the error on nearly all sequences. A third iteration does not reduce the error further.

7.5.1.2 Comparison with State-of-the-art

In our experiments, we consider two sources for the motion capture data, namely HumanEva-I and the CMU motion capture dataset.

HumanEva-I Dataset. We first use the entire 49K 3D poses of the training data as motion capture data and compare our approach with the state-of-the-art methods [KG14, WWL+14, RDG13, SSQTMN13, SSRA+12, BS10]. Although the training data for 2D pose estimation and 3D pose data are from the same dataset, our approach considers them as two different sources and does not know the 3D pose for a training image. We report the 3D pose error for each sequence in Table 7.1 and the average

2D Pose Estimation Error on HumanEva-I dataset							
Methods	**Walking (A1, C1)**			**Jogging (A2, C1)**			**Average**
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	
Dantone *et al.* [DLGVG14]†	9.94	8.53	12.04	12.54	9.99	12.37	10.90
Wang *et al.* [WL13]	17.47	17.84	21.24	16.93	15.37	15.74	17.43
Desai *et al.* [DR12]	10.44	9.98	14.47	14.40	10.38	10.21	11.65
Yang *et al.* [YR11]	11.83	10.79	14.28	14.43	10.49	11.04	12.14
(a) 2D Pose Refinement (MoCap from HumanEva-I dataset)							
(i) Refinement with MoCap Priors							
Iteration-I	6.96	6.08	9.20	9.80	7.23	8.71	8.00
Iteration-II	**6.47**	**5.50**	**8.54**	**9.40**	**6.79**	**7.99**	**7.45**
(ii) Refinement with Projection of 3D Estimated Pose							
Iteration-I	7.43	6.14	10.25	9.91	7.97	10.44	8.69
Iteration-II	6.78	6.16	9.10	9.64	6.99	7.83	7.75
(iii) Refinement with Projection of 3D Estimated Pose (Rigid Alignment)							
Iteration-I	5.15	4.58	6.88	6.41	5.91	6.24	5.86
Iteration-II	4.79	4.47	6.10	6.14	5.53	5.61	5.44
(b) 2D Pose Refinement (MoCap from CMU dataset)							
(i) Refinement with MoCap Priors							
Iteration-I	7.62	6.26	10.99	11.14	8.58	9.93	9.08
Iteration-II	7.12	5.99	10.64	10.79	8.24	9.42	8.70
(ii) Refinement with Projection of 3D Estimated Pose							
Iteration-I	8.42	7.11	11.48	12.11	9.64	11.24	10.0
Iteration-II	7.98	6.51	11.02	11.67	9.40	10.17	9.45
(iii) Refinement with Projection of 3D Estimated Pose (Rigid Alignment)							
Iteration-I	6.59	7.26	9.01	9.02	9.91	8.92	8.45
Iteration-II	6.28	6.83	8.98	8.98	9.61	8.44	8.18

Table 7.3: Comparison with state-of-the-art approaches for 2D pose estimation error (pixels) after refinement. **(a)** and **(b)** explore the 2D pose estimation error after performing pose refinement using MoCap from HumanEva-I dataset and CMU dataset respectively. **(i)** represents refinement using MoCap priors (Section 7.4.2), **(ii)** show results of 2D pose after projecting final 3D estimated pose while **(iii)** corresponds to 2D pose resulted through projection of 3D estimated pose after rigid alignment. † denotes the method which corresponds to our initial 2D pose estimation.

error in Table 7.2. While there is no method that performs best for all sequences, our approach outperforms all other methods in terms of average 3D pose error. The approaches [KG14, BS10] achieve a similar error, but they rely on stronger assumptions. In [KG14] the ground-truth is used to compute a 3D bounding volume and the inference requires around three minutes per image since the approach uses a 3D PSM. The

Component wise Execution Run Time in Seconds		
Components	**Run Time**	**Details**
Feature sets extraction	27.0 sec.	144 virtual cameras are used to develop 2D feature sets. It is a pre-processing step.
kd-tree development	14.0 sec.	For HumanEva-I dataset with 49K number of frames. It is also a pre-processing step.
Total Time	41.0 sec.	
2D pose estimation	10.0 sec.	Pyramid of 6 scales is used and scale factor is kept 0.85.
3D pose retrieval †	0.12 sec.	$0.024 \times 5 = 0.12$ sec., a cumulative execution time for 256 nearest neighbors retrieval on the basis of all 5 joint sets $\mathcal{J}_s$.
Projection $\mathcal{M}_s$ †	4.75 sec.	$0.95 \times 5 = 4.75$ sec., a cumulative execution time for all 5 joint sets $\mathcal{J}_s$ on projection of 256 nearest neighbors onto image.
2D pose refinement †	3.10 sec.	Only binary potentials are involved for pose refinement (7.12).
3D pose estimation †	0.15 sec.	Time for 3D pose estimation (7.9).
Total Time	18.12 sec.	

Table 7.4: Execution run time in seconds for each component involved in proposed method for HumanEva-I dataset. The image size is 640×480 pixels. The execution time is measured on a 12-core 3.2GHz Intel processor. † denotes the components which also participate in second iteration.

first iteration of our approach takes only 18.12 seconds per image (see Table 7.4) and additional 8.12 seconds for a second iteration. In [BS10] background subtraction is performed to obtain the human silhouette, which is used to obtain a tight bounding box. The approach also uses 20 joints instead of 14, which therefore results in a different 3D pose error. We therefore use the publicly available source code [BS10] and evaluate the method for 14 joints and provide the human bounding box either from ground-truth data (GT-BB) or from our 2D pose estimation (Est-BB). The results in Table 7.1 and Table 7.2 show that the error significantly increases for [BS10] when the same skeleton is used and the human bounding box is not given but estimated.

CMU Motion Capture Dataset. In contrast to the other methods, we do not assume that the images are annotated by 3D poses but use motion capture data as a second training source. We therefore evaluate our approach using the CMU motion capture dataset [CMU14] for our 3D pose retrieval. We use one third of the CMU dataset and downsample the CMU dataset from 120Hz to 30Hz, resulting in 360K 3D poses. Since the CMU skeleton differs from the HumanEva skeleton, the skeletons are mapped to the HumanEva dataset by linear regression. The results are shown in Table 7.1(b)

The Impact of MoCap Data on 3D Pose Estimation							
MoCap data	**Walking (A1, C1)**			**Jogging (A2, C1)**			**Average**
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	
(a) HumanEva	40.1	33.1	47.5	48.6	43.6	40.0	42.1
(b) HumanEva\Walking	70.5	60.4	86.9	46.5	40.4	38.8	57.3
(c) HumanEva-Retarget	59.5	43.9	63.4	61.0	51.2	55.7	55.8
(d) CMU	54.5	54.2	64.2	76.2	74.5	58.3	63.6

Table 7.5: Impact of MoCap data. **(a)** MoCap from HumanEva dataset. **(b)** MoCap from HumanEva dataset without walking sequences. **(c)** MoCap from HumanEva dataset but skeleton is retargeted to CMU skeleton. **(d)** MoCap from CMU dataset. The average 3D pose error (mm) is reported for the HumanEva-I dataset with one iteration.

and Table 7.2(b). As expected the error is higher due to the differences of the datasets, but the error is still low in comparison to the other methods.

To analyze the impact of the motion capture data more in detail, we have evaluated the pose error for various modifications of the data in Table 7.5. We first remove the walking sequences from the motion capture data. The error increases for the walking sequences since the dataset does not contain poses related to walking sequences any more, but the error is still comparable with other state-of-the-art methods (Table 7.1 and Table 7.2). The error for the jogging sequences actually decreases since the dataset contains less poses that are not related to jogging. In order to analyze how much of the difference between the HumanEva and the CMU motion capture data can be attributed to the skeleton, we mapped the HumanEva poses to the CMU skeletons. As shown in Table 7.5(c), the error increases significantly. Indeed, over 60% of the error increase can be attributed to the difference of skeletons. In Table 7.3 we also compare the error of our refined 2D poses with other approaches. We report the 2D pose error for [DLGVG14], which corresponds to our initial 2D pose estimation as described in Section 7.3. In addition, we also compare our method with [YR11, WL13, DR12] using publicly available source codes. The 2D error is reduced by pose refinement using either of the two motion capture datasets and is lower than for the other methods. In addition, the error is further decreased by a second iteration. We also report the error for 2D pose resulted through projection of 3D estimated pose with and without rigid alignment. Some qualitative results are shown in Figure 7.7.

7.5.2 Evaluation on Human3.6M Dataset

The protocol originally proposed for the Human3.6M dataset [IPOS14] uses the annotated bounding boxes and the training data only from the action class of the test data.

Comparison on the Human3.6M dataset								
Methods	**Directions**	**Discussion**	**Eat**	**Greet**	**TalkOnPhone**	**Posing**	**Purchase**	**Sit**
H3.6M	88.4	72.5	108.5	110.2	97.1	81.6	107.2	119.0
H3.6M + 2D GT	60.0	54.7	71.6	67.5	63.8	61.9	55.7	73.9
H3.6M + 3D GT	66.2	57.8	98.8	84.5	79.6	58.2	100.7	115.8
CMU	102.8	80.4	133.8	120.5	120.7	98.9	117.3	150.0

Methods	**SitDown**	**Smoking**	**TakingPhoto**	**Wait**	**Walk**	**WalkwithDog**	**WalkTogether**
H3.6M	170.8	108.2	142.5	86.9	92.1	165.7	102.0
H3.6M + 2D GT	110.8	78.9	96.9	67.9	47.5	89.3	53.4
H3.6M + 3D GT	162.1	97.2	119.2	73.4	88.5	159.1	99.8
CMU	182.6	135.6	140.1	104.7	111.3	167.0	116.8

Table 7.6: The average 3D pose error (mm) on the Human3.6M dataset for all actions of subject S11.

Comparison on the Human3.6M dataset	
Methods	**3D Pose Error (mm)**
Kostrikov *et al.* [KG14]	115.7
Bo *et al.* [BS10]	117.9
(i) Our Approach (MoCap from Human3.6M dataset)	
(a) H3.6M (Iteration I)	**108.3**
(b) H3.6M + 2D GT (Iteration I)	70.5
(c) H3.6M + 3D GT (Iteration I)	95.2
(ii) Our Approach (MoCap from CMU dataset)	
CMU (Iteration I)	124.8

Table 7.7: Comparison on the Human3.6M dataset. **(a)** 2D pose estimated as in Section 7.3 **(b)** 2D pose from ground-truth. **(c)** MoCap dataset includes 3D pose of subject S11.

Since this protocol simplifies the task due to the small pose variations for a single action class and the known scale, a more realistic protocol has been proposed in [KG14] where the scale is unknown and the training data comprises all action classes. We follow the protocol [KG14] and use every 64th frame of the subject S11 for testing. Since the Human3.6M dataset comprises a very large number of training samples, we increased the number of regression trees to 30 and the number of mixtures of parts to $c = 40$, where each tree is trained on 10K randomly selected training images. We use the same *3D pose error* for evaluation and perform the experiments with 3D pose data from Human3.6M and the CMU motion capture dataset.

In the first case, we use six subjects (S1, S5, S6, S7, S8 and S9) from Human3.6M and eliminate very similar 3D poses. We consider two poses as similar when the average Euclidean distance of the joints is less than $1.5mm$. This resulted in 380K 3D poses. In the second case, we use the CMU pose data as described Section 7.5.1.2. The results

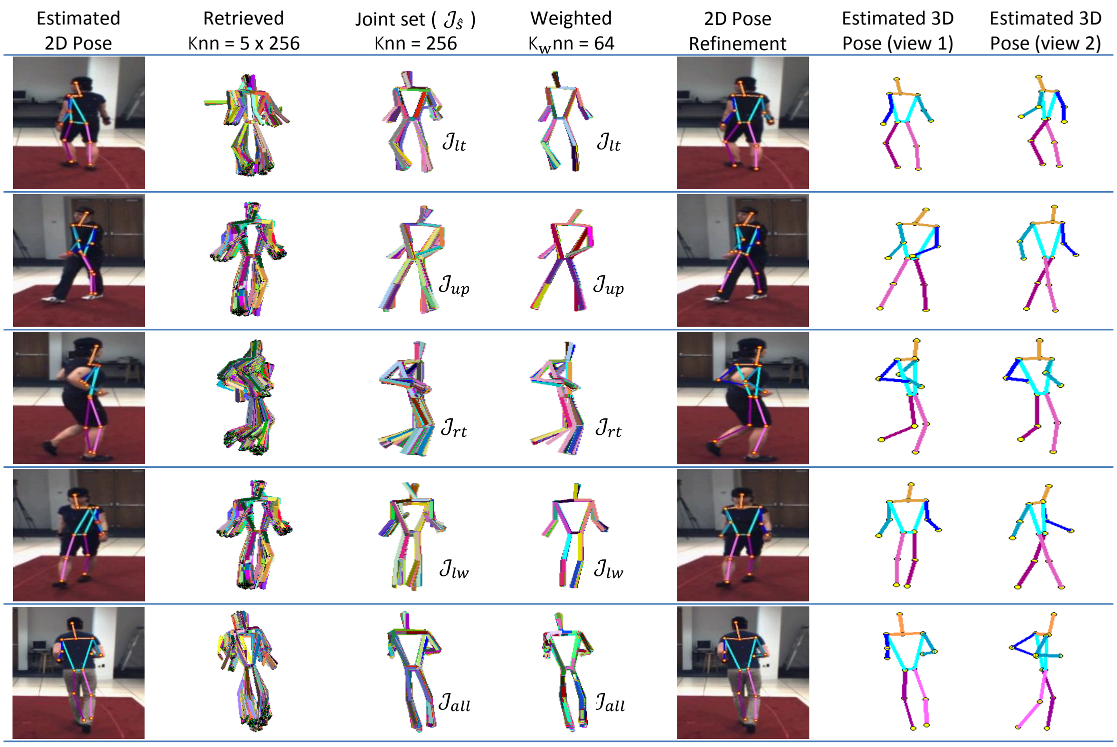


Figure 7.7: Five examples from HumanEva-I. From left to right: estimated 2D pose $\mathbf{x}$ (Section 7.3); retrieved 3D poses from all joint sets (Section 7.4.1); retrieved 3D poses from inferred joint set $\mathcal{J}_{\hat{s}}$ (Section 7.4.2); retrieved 3D poses weighted by $w_{k,\hat{s}}$ (7.14); refined 2D pose $\hat{\mathbf{x}}$ (7.12); estimated 3D pose $\widehat{\mathbf{X}}$ (7.9) shown from two different views.

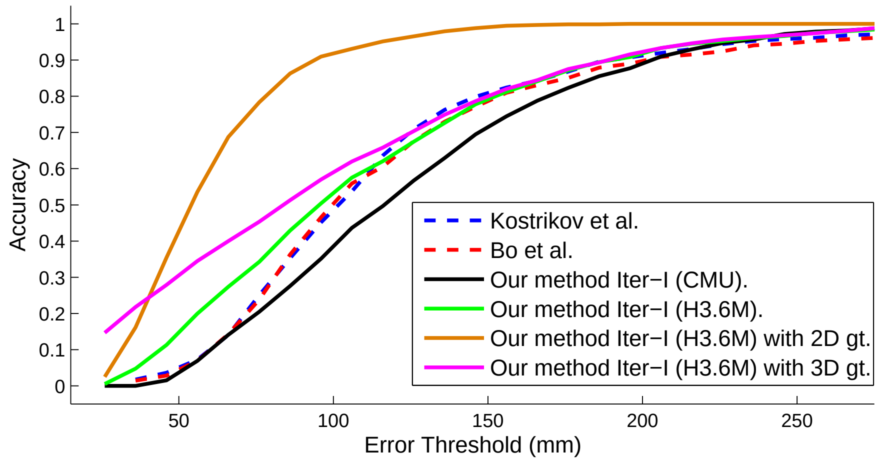


Figure 7.8: Comparison on the Human3.6M dataset.

are reported in Tables 7.7 and 7.6. Table 7.7 shows that our approach outperforms [KG14, BS10].

Although a second iteration does not reduce the error on this dataset, our approach

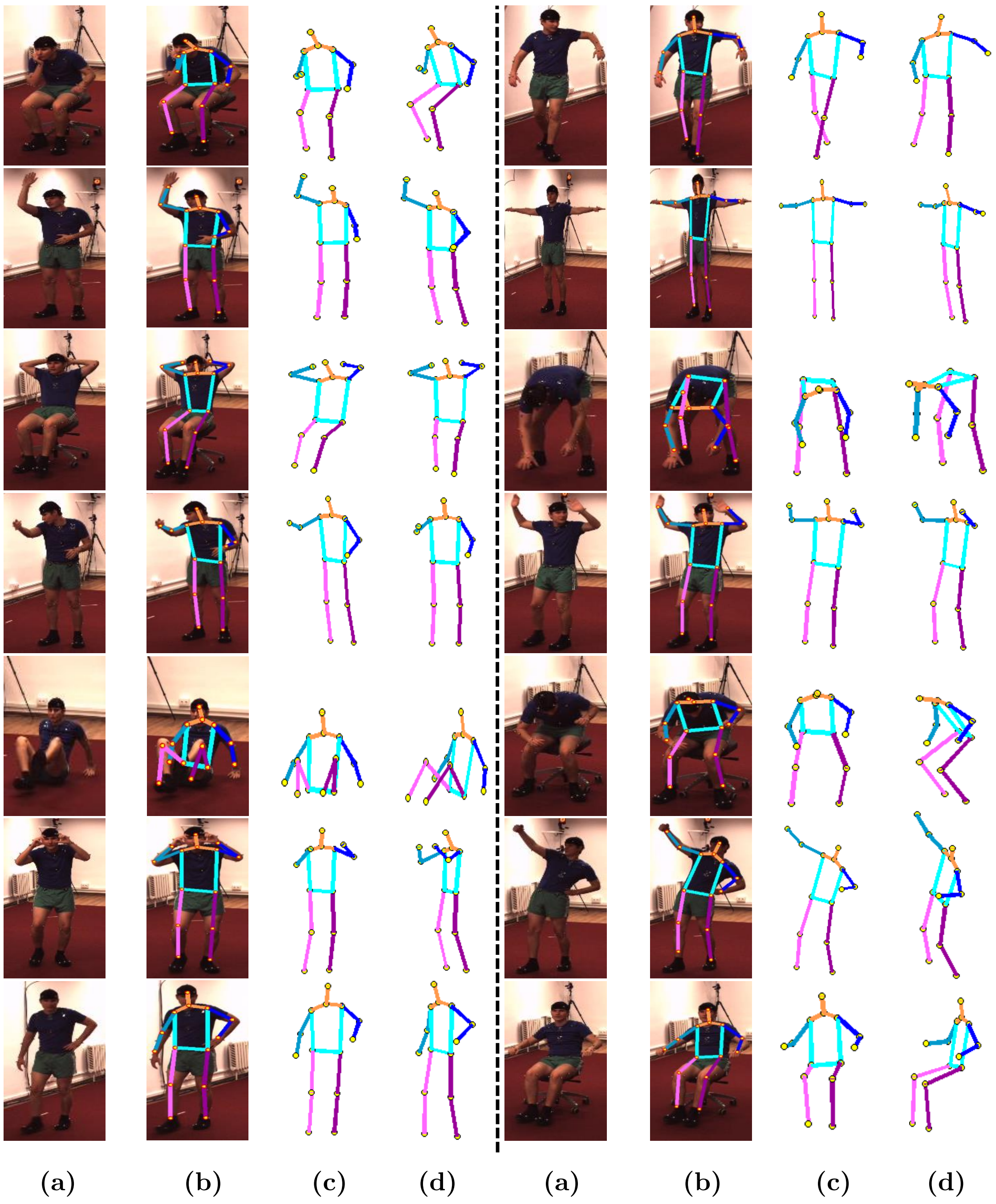


Figure 7.9: A few qualitative results from Human3.6M dataset [IPOS14]: **(a)** represents input images, **(b)** shows refined 2D poses while **(c)** and **(d)** correspond to estimated 3D poses from two different views.

outperforms the other approaches. Figure 7.8 provides a more detailed analysis and shows that more joints are estimated with an error below 100mm in comparison to the other methods. When using the CMU motion capture dataset, the error is again higher due to differences of the datasets but still competitive.

We also investigated the impact of the accuracy of the initially estimated 2D poses.

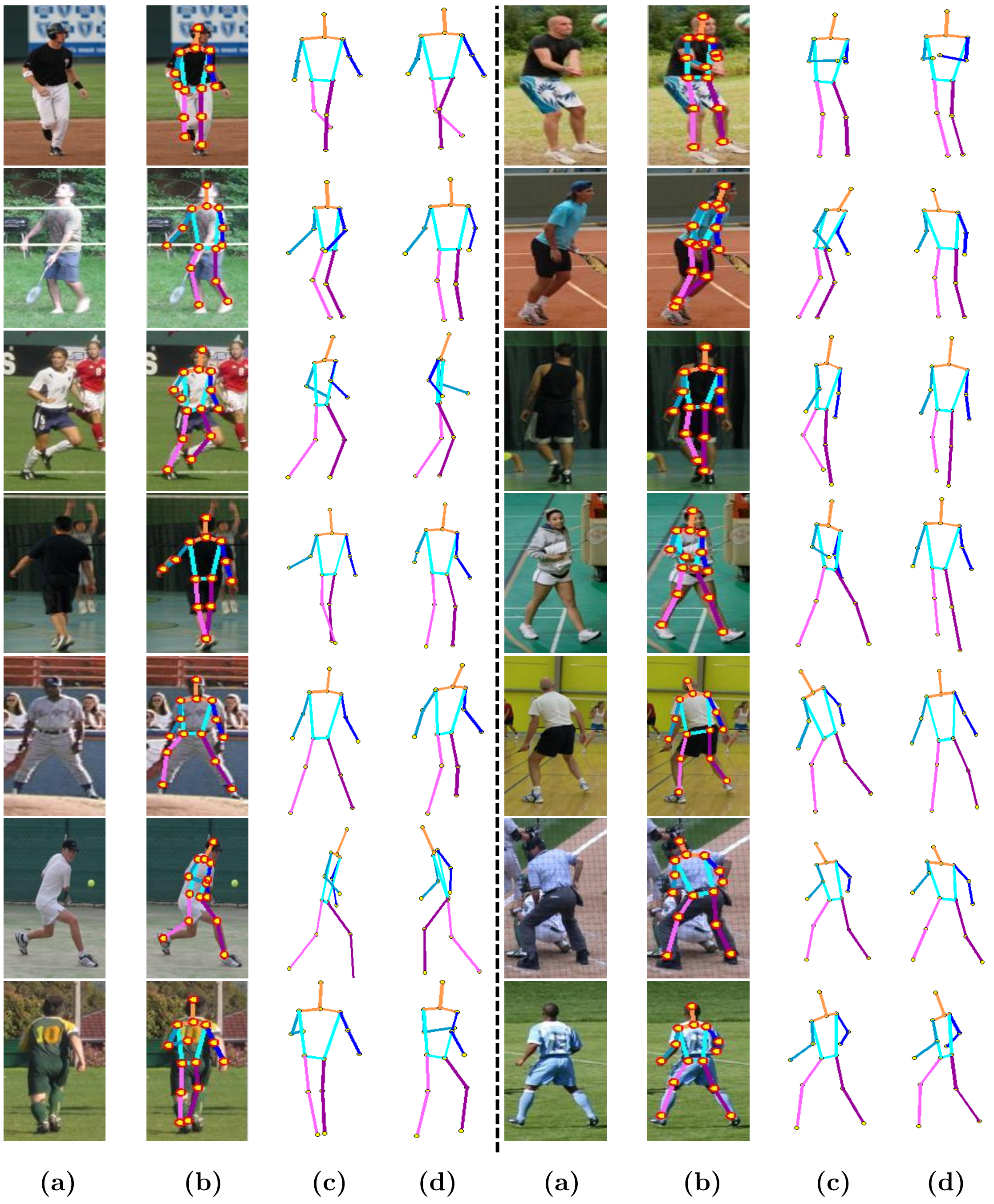


Figure 7.10: A few qualitative results from Leeds Sports pose dataset [JE10]: **(a)** represents input images, **(b)** shows refined 2D poses while **(c)** and **(d)** correspond to estimated 3D poses from two different views.

If we initialize the approach with the 2D ground-truth poses, the 3D pose error is drastically reduced as shown in Table 7.7(b) and Figure 7.8. This indicates that the 3D pose error can be further reduced by improving the used 2D pose estimation method. In Table 7.7(c), we also report the error when the 3D poses of the test sequences are added to the motion capture dataset. While the error is reduced, the impact is lower compared to accurate 2D poses or differences of the skeletons (CMU). The error for each action class is given in Table 7.6.

We have found that our approach performs poorly on tightly cropped images since the used 2D pose estimation approach (Section 7.3) requires a minimum distance from a joint to the image border. Furthermore, differences of the skeleton structure between datasets have a significant impact on the accuracy and the error also increases when the dataset does not contain poses related to the test sequences.

7.5.3 Qualitative Results

We present some qualitative results for the Human3.6M dataset [IPOS14] as well as Leeds Sports pose dataset [JE10]. Human3.6M dataset contains images captured in an indoor environment while Leeds Sports pose dataset consists of realistic images taken from the internet. For experiments on Leeds Sports pose dataset we train our regression forests and pictorial structure model using 1000 training images provided with the dataset, and use CMU motion capture dataset to develop our motion capture database. A few examples of resulting 3D pose estimates for both datasets are shown in Figure 7.9 and Figure 7.10, respectively. As evident in Figure 7.9 and Figure 7.10, our approach shows very good performance even for highly articulated poses, and also for images captured in unconstrained environments.

7.6 Conclusion

In this paper, we have presented a novel dual-source approach for 3D pose estimation from a single RGB image. One source is a motion capture dataset with 3D poses and the other source are images with annotated 2D poses. In our experiments, we have shown that our approach achieves state-of-the-art results when the training data are from the same dataset, although our approach makes less assumptions on the training and test data. Our dual-source approach also allows to use two independent sources and still executes competitive results.

Part III

Quadruped Motions: Retrieval and Reconstruction

8 Retrieval and Reconstruction of Quadruped Motions

Measure what is measurable, and make measurable what is not so.

Galileo Galilei

8.1 Introduction

This chapter focus on retrieval and reconstruction of the quadruped motions. The 3D capturing of motions especially for human has become a standard technique in multitude data-driven applications. There are many ways to capture motion data such as, mechanical, magnetic, optical and inertial sensor based systems *etc.* These systems are nowadays available in all prices, starting from the consumer electronics (e.g. Kinect, WiiMote) up to the professional optical systems like Vicon or Giant. All these technologies have their strengths and weaknesses—an overview is given in [MHK06]. The increasing amount of motion capture data allows for many new applications in the field of computer animation, human computer interaction, sport sciences, medicine and biomechanics as described in Chapter 1. Due to high profile usability, the motion capture data gets importance not only for the human motions but also for the quadruped motions *i.e.*: the 3D motion capture data of horses are used in research and the understanding of clinical treatment [HLR+10]; the clinical biomechanics are used for the treatment of the domestic animals, clinical gait analysis and the medical rehabilitation *etc.* On one hand, the motion analysis for the quadrupeds has become a powerful tool to record the movement patterns during gait and other exercises in clinical environments. On the other hand, the quadruped MoCap data can be interesting for games, if one considers animation of the non-humanoid characters [YAH10, VHKK12]. As most of the techniques in computer animation are developed to handle the human motion data, we adjust several well-known techniques from computer animation to leverage with the quadrupedal motion capture data in order to cover this gap, and the results are reported by conducting a series of experiments in this work.

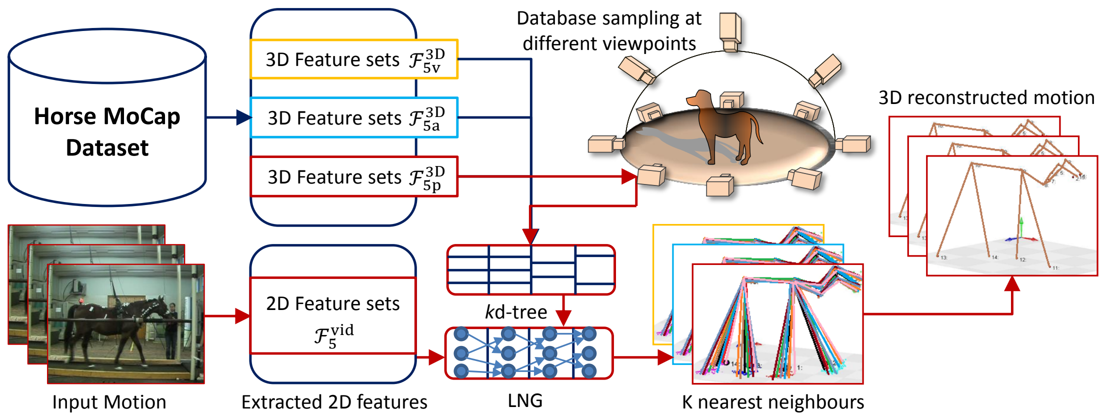

Horse MoCap Dataset
3D Feature sets $\mathcal{F}^{3D}_{5v}$
3D Feature sets $\mathcal{F}^{3D}_{5a}$
3D Feature sets $\mathcal{F}^{3D}_{5p}$
Database sampling at different viewpoints
3D reconstructed motion
2D Feature sets $\mathcal{F}^{vid}_{5}$
kd-tree
Input Motion
Extracted 2D features
LNG
K nearest neighbours

Figure 8.1: System overview: The retrieval and reconstruction of the motion sequences are performed on the basis of different types of 3D and 2D feature sets. The *red* color corresponds to those components and the feature sets that are involved in the reconstruction process.

First, we discuss some aspects that have to be considered during recording animal motions (Section 8.2). Second, we adapt the retrieval techniques to integrate with the quadruped data. Here, the crucial step is to define suitable and meaningful feature sets for the quadruped motions. This will be discussed in Section 8.3. Finally, we introduce a framework for motion reconstruction of the quadruped animals from the video data by utilizing some prior knowledge obtained from the motion capture database. We have reshaped the methodology for reconstruction of the human motion from the video data as mentioned in Chapter 4 and map that technique on reconstruction of the quadruped motions with some significant modifications. Section 8.4 will elaborate this in detail. At the end, results and analysis are presented in Section 8.5 and conclusion in Section 8.6.

8.2 Quadruped MoCap Dataset

In this section, we present some details on the recording environment of the motion capture data. We use three-dimensional kinematic data captured from five mature horses. The MoCap data is represented in a standard right-handed Cartesian coordinate system and is recorded using 10 digital infrared cameras (Eagle Digital Realtime System) at 120 Hz on the treadmill. Motion capturing in the standardized environment has several benefits such as, the recording on the treadmill is possible which establishes the more accurate measurement setups and provides more opportunities to capture various gait by just controlling the speed of the treadmill which plays an essential role in recording the different gait measurements.

Marker No.	**Marker name**	**Placement of the markers**
1	ChistaFacialisL	Left side of the facial crest.
2	ChistaFacialisR	Right side of the head.
3	C1L	Left side of the first cervical vertebra.
4	C1R	Right side of the first cervical vertebra.
5	C3L	Left side of the third cervical vertebra.
6	C3R	Right side of the third cervical vertebra.
7	C6L	Left side of the sixth cervical vertebra.
8	C6R	Right side of the sixth cervical vertebra.
9	Withers	Highest point of the withers.
10	Sacrum	Highest point of the sacral region.
11	Rightfore	Lateral side of the right front hoof.
12	Leftfore	Lateral side of the left front hoof.
13	Righthind	Lateral side of the right hind hoof.
14	Lefthind	Lateral side of the left hind hoof.
15	*Forehead (Virtual)*	*By taking mean of markers no. 1 and 2.*

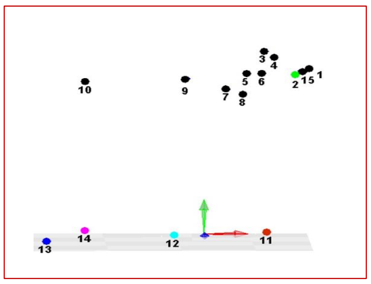
Table 8.1: List of the motion capture markers as well as the details about their placement on the body of the performing horse.

8.2.1 Marker Setup

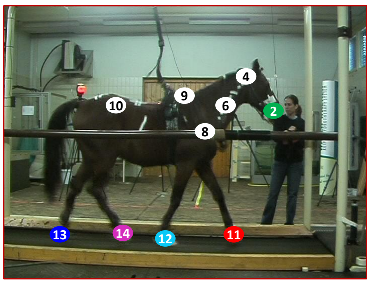
For the MoCap recording, the retro reflective skin markers are attached to each horse using adhesive tape. The marker setups can be varied according to the recording and measurements objectives. In a basic MoCap setup for horses, generally seven markers are required to capture the whole body motion. The first marker is normally placed on the head, then two on the trunk and the four on the hooves. However, the number of markers can be increased, when the recording and measurement purpose is more complex and requires more detailed motion capturing. In addition, the marker setups may vary between different subjects due to the size variations. In our case, the markers are placed on the head (left and right crista facialis), on the highest point of the withers, sacrum and the lateral side of each hoof in order to identify the motion cycles. Since the MoCap data used in this work are originally recorded in a clinical setup where the research is focused on the neck movement in different types of gait, that's why the additional markers are attached along the vertebrae of the horses' neck. A list of all these markers is given in Table 8.1. The marker placement on the animal's skin can be challenging since the anatomical landmarks are not always easily detectable on different animals as well as due to the skin artifacts. This may lead towards the marker displacement which always needs to be considered during the motion capturing and the clinical investigations.

(a) MoCap environment

(b) Marker setup



(c) 3D markers' visualization



1: CristaFacialisL (occluded)
2: CristaFacialisR
3: C1L (occluded)
4: C1R
5: C3L (occluded)
6: C3R
7: C6L (occluded)
8: C6R
9: Withers
10: Sacrum
11: Right Fore
12: Left Fore
13: Right Hind
14: Left Hind
15: Forehead (virtual)

(d) Marker setup details

Figure 8.2: The motion capturing environment and the setup for the horse motions: **(a)** An example of motion recording environment; **(b)** The marker setup on the horse performing different gait sequences on the treadmill; **(c)** The 3D representation of the quadruped markers that generate final quadruped motion capture dataset; **(d)** Marker setup with marker IDs. The *colored* circles show those markers which are selected to develop different feature sets.

8.2.2 Motion Capture Dataset

Under the recording conditions described above, the motion sequences of five horses are recorded. Each horse performed at least three trials (each of 10 seconds) of two motion styles walk and trot.

Walk. The walk is a four-beat gait and with a slow pace, the horse always have one foot raised and the other three feet on the ground, except for a brief moment when the weight is being transferred from one foot to another. A horse moves its head and neck in a slight up and down motion that helps to maintain balance. [Wik13].

Trot. The trot is a two-beat gait and the horse moves its legs simultaneously in diagonal pairs. In contrast to walking motion, it is a very stable gait and the horse need not to make major balancing motions with its head and neck. [Wik13].

As a result, we have a dataset that consists of 30 motion trials with a varying number of motion cycles. The total amount of the dataset sums up to 36,000 frames, sampled at 120 Hz which corresponds to five minutes of the motion capture data. We denote this full MoCap dataset as $\mathcal{DB}_{\text{quad}}$. For our experiments, we work with various down-sampled versions of this dataset. If this is the case, the upper index denotes the sampling frequency *e.g.* for the video-based reconstruction, we have down-sampled the MoCap database $\mathcal{DB}_{\text{quad}}$ at sampling rate 25 Hz and refer it as $\mathcal{DB}_{\text{quad}}^{25\text{Hz}}$ which contains overall $7,500$ frames.

It is important to note that due to the relatively sparse marker setup, 14 markers in the quadruped MoCap dataset as compared to 42 markers in the HDM05 [MRC+07] and CMU [CMU14] MoCap datasets, it is not possible to fit a suitable skeleton to the recorded marker data. Thus, we represent the skeleton model consisting of 15 marker positions with one virtual marker position. A few exemplary images for the MoCap environment, the marker setup with marker IDs and the 3D visualization of the markers are presented in Figure 8.2.

8.3 Motion Retrieval

For search and retrieval of the similar motion segments, the dataset is a crucial component in all data-driven methods. In this chapter, we are dealing with the horse MoCap dataset which contains sparse marker setup with 14 markers and have no any skeleton parameterizations. While in skeleton representations of the human motion data, the root node is located between the hip joints, but for the quadruped motions other different choices to locate the root node can be possible. Huang *et al.* [HHL13] employ 35 joints with 61 degrees of freedom and set the spine of the horse as the root node. The authors locate the root node close to the hind legs and this choice is very close to the skeleton based representation of the human motion data. In contrast, we consider the marker *Withers* (No. 9 in our marker set) to be the root marker. This choice of root location is motivated by the observation that the root node in human representations is very close to the whole body's center of mass. For quadrupeds, the center of mass is more close to the forelegs [BOS00, NKMC09]. Thus, with the marker *Withers* as a root node, we obtain a normalized pose representation of the quadrupeds with more detailed characteristics. For search and retrieval, we construct various 3D and 2D feature sets extracted either from the motion capture data or from the video data as reported in Table 8.2. We develop these feature sets on the basis of the four end effectors and the head, because such a feature set has a simple computation, low dimensions and still meaningful in describing the poses [KTWZ10] even in case of the quadrupeds. We denote the joints involved in the horse skeleton with $\mathcal{J}$ while the joints involved in the feature set are represented by $\mathcal{J}_{\mathcal{F}}$.

8.3.1 3D Feature Sets

We develop three different kinds of 3D feature sets based on the marker positions, velocities and accelerations. We extract the feature set $\mathcal{F}_{5\text{p}}^{3\text{D}}$ where the positions of the four hooves and the head are used to describe a quadrupedal pose in a relative coordinate system. In the horse MoCap dataset, we have no any skeletal representation and as a result we perform the process of pose normalization on the marker positional data directly—the positions of the all markers are normalized relative to the *Withers* marker (root node), after rotating all marker positions around the y axis such that the *Sacrum* marker (No. 10) is moving in the x-y-plane. We also develop the feature sets that consist of the velocities and the accelerations of the markers represented by $\mathcal{F}_{5\text{v}}^{3\text{D}}$ and $\mathcal{F}_{5\text{a}}^{3\text{D}}$ respectively. The idea is to have a similarity search that might be based on the inputs from other sensor types such as acceleration sensors. These sensors have been used to reconstruct full body human poses [TZK+11]. With such experiments, in principle, we want to evaluate the possibilities to reconstruct the quadruped motions based on such sensor configurations.

8.3.2 2D Feature Sets

In order to reconstruct the motion sequences from the 2D input video either synthetic or the real video, we need to search into the MoCap database for the similar poses based on the 2D feature sets extracted from the input signals. We introduce the 2D feature sets which are derived from the horse MoCap data as well as from the video input data. These feature sets from the different domains have been made comparable in order to accomplish the cross model retrieval scenario between 2D input signals and the 3D MoCap database. In this context, we sample the feature sets from the MoCap dataset at as many as needed view directions to find similar poses from the MoCap dataset, without having any information about the actual view direction of the camera.

Motion Capture Data. We extract the 2D feature sets $\mathcal{F}_5^{\text{syn}}$ through orthographic projection of the 3D feature set $\mathcal{F}_{5\text{p}}^{3\text{D}}$ onto 2D image plane at different view directions—the azimuth angles (0–10–350) degrees with step size 10 degree and the elevation angles (0–10–90) degrees with step size 10 degree. We translate these 2D feature sets so that they locate their origin at center of mass (*Withers* marker) in order to be comparable with later described video-based 2D feature sets. On the basis of these 2D feature sets, we search into MoCap dataset for the closest instances through a kd-tree.

Video Data. In case of video data, the 2D feature sets $\mathcal{F}_5^{\text{vid}}$ are detected and tracked with the help of SURF and MSER feature detection techniques (Section 4.3.2). Under

Feature sets	Type	Description of the feature sets
$\mathcal{F}_{5\mathrm{p}}^{\mathrm{3D}}$	3D	This feature set is developed on the basis of the normalized positions of the hooves markers and the head marker.
$\mathcal{F}_{5\mathrm{v}}^{\mathrm{3D}}$	3D	It is based on the derived velocities of the hooves and the head markers.
$\mathcal{F}_{5\mathrm{a}}^{\mathrm{3D}}$	3D	It represents the derived accelerations of the hooves and the head markers.
$\mathcal{F}_{5}^{\mathrm{syn}}$	2D	This feature set represents the normalized 2D positions of the hooves and the head markers obtained through projection of the 3D MoCap data onto the 2D image plane.
$\mathcal{F}_{5}^{\mathrm{vid}}$	2D	It is based on the normalized 2D positions of the hooves and the head markers extracted from the video input data.

Table 8.2: The details of different types of feature sets (3D and 2D) developed in order to retrieve K nearest neighbors from the quadrupedal MoCap dataset.

scaled orthographic camera model with projection matrix $\mathcal{M}$, we estimate the unknown scaling factor from the first few frames of the MoCap dataset and the corresponding video frames as discussed in Section 4.3.2. To be comparable with synthetic 2D feature sets $\mathcal{F}_5^{\mathrm{syn}}$, we normalize video-based 2D feature sets $\mathcal{F}_5^{\mathrm{vid}}$ by translating them to their center of mass.

8.3.3 Knn Search

We perform two types of Knn search: First, the similar poses are found in the motion capture database. After computing the feature sets for all frames of the motion database, the K nearest neighbors can efficiently be retrieved by searching through a kd-tree. Second, we search for the K most similar motion sequences using a technique called *lazy neighborhood graph* (LNG). It is a graphical structure, where all K nearest neighbors of the query frames are considered as nodes. An edge between a pair of nodes is inserted whenever the indices allow for a connection based on the step size conditions. The result is a single source shortest path problem on a directed, acyclic graph, where every shortest path in the graph corresponds to a warping path between the query motion and a motion segment in the database.

8.4 Reconstruction from Video

For 3D motion reconstruction, we exploit the pre-existing knowledge embedded in the MoCap database to lift from 2D to 3D. To this end, we have nearest neighbors as described in Section 8.3, on which basis we perform a data-driven energy minimization. We adapt the problem formulation as in Section 4.4 and modify it according to the situation when we have neither skeletal information nor joint angle configurations. As we have lack of this information in our MoCap dataset, we perform the 3D motion reconstruction pose by pose by just employing the joint positions $\mathcal{X}_t = \{\mathbf{X}_{t,\mathsf{k}} | \mathsf{k} = 1, \ldots, \mathsf{K}\}$ directly in Euclidean space. Having in hand K nearest neighbors, a 3D pose model is developed in low-dimensional PCA subspace of the joint positions $\mathcal{X}$ of K nearest neighbors.

$$\widetilde{\mathbf{X}}_t = \sum_{v=1}^{V} \mathsf{c}_{t,v} \mathsf{b}_{t,v} + \mu_t \tag{8.1}$$

where $\widetilde{\mathbf{X}}_t$ is the 3D synthesized pose which is the linear combination of a set of V bases $\mathsf{B}_t = \{\mathsf{b}_{t,1}, \ldots, \mathsf{b}_{t,V}\}$, μ_t is the mean pose and c_t are the basis coefficients at frame t. From local modeling towards global modeling, we only deal with the low dimensional space and as a result we make the process of data-driven optimization roughly realtime with low computations. The energy function for the 3D reconstruction is,

$$\widehat{\mathbf{X}} = \arg\min_{\widetilde{\mathbf{X}}} (w_p E_p + w_c E_c + w_s E_s), \tag{8.2}$$

minimized using Levenberg-Marquardt based nonlinear optimizer. The notations w_p, w_c and w_s are the associated weights with the energy terms having values $w_p = 1$, $w_c = 0.75$ and $w_s = 0.05$. Each energy term is normalized by a normalization factor ℓ_t.

The energy, E_p, maps MoCap prior information to synthesized pose as,

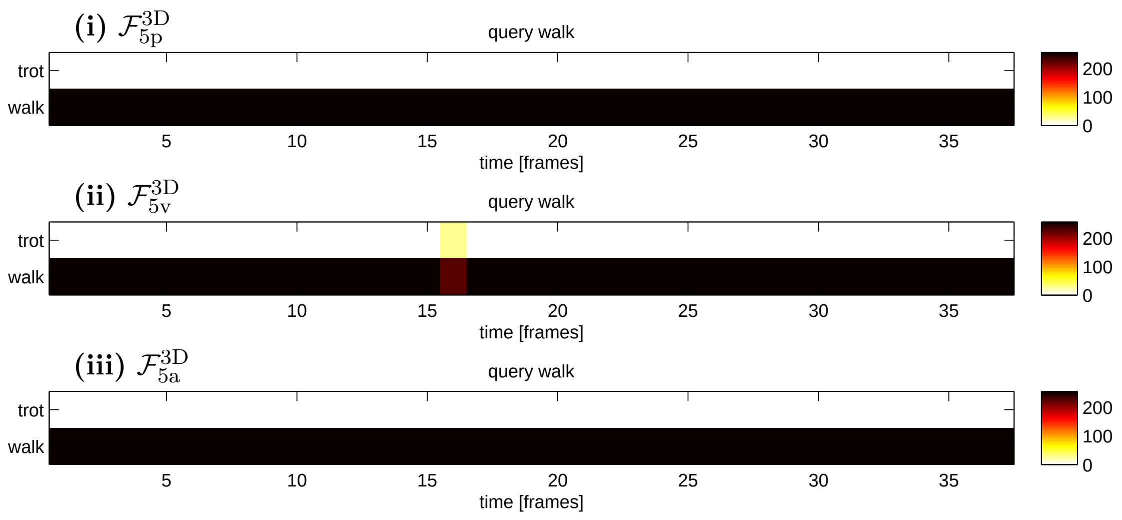
$$E_p(\widetilde{\mathbf{X}}) = \frac{1}{\sqrt{\ell_t}} \cdot \|(\widetilde{\mathbf{X}}_t - u_t)^T \Lambda_t^{-1} (\widetilde{\mathbf{X}}_t - u_t)\|^2, \tag{8.3}$$

where u_t is the mean vector of Knn and the Λ_t^{-1} is the inverse of the covariance matrix at frame t.

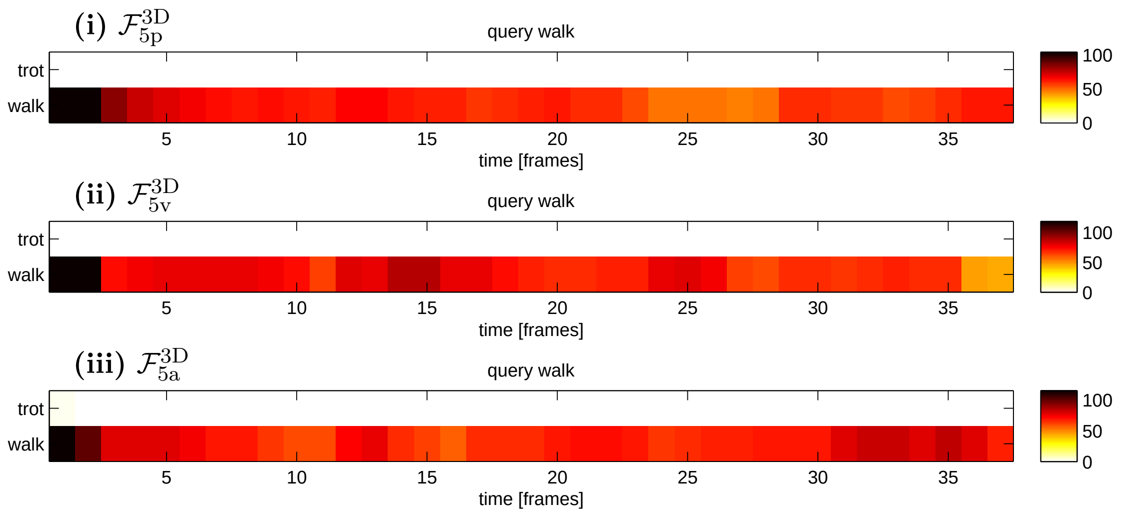
The second energy, E_c, computes the projection error between the 3D-2D feature sets as,

$$E_c(\widetilde{\mathbf{X}}) = \frac{1}{\sqrt{\ell_t}} \cdot \sqrt{\sum_{i \in \mathcal{J}_{\mathcal{F}}} \|\mathcal{M}_t \widetilde{X}_{t,i} - x_{t,i}\|^2}, \tag{8.4}$$

where $\widetilde{X}_{t,i}$ is the i^{th} joint's 3D position of the synthesized pose, and $x_{t,i}$ is the i^{th} joint's 2D position at current frame t.



(a) K nearest neighbors retrieved through the Knn-based method.



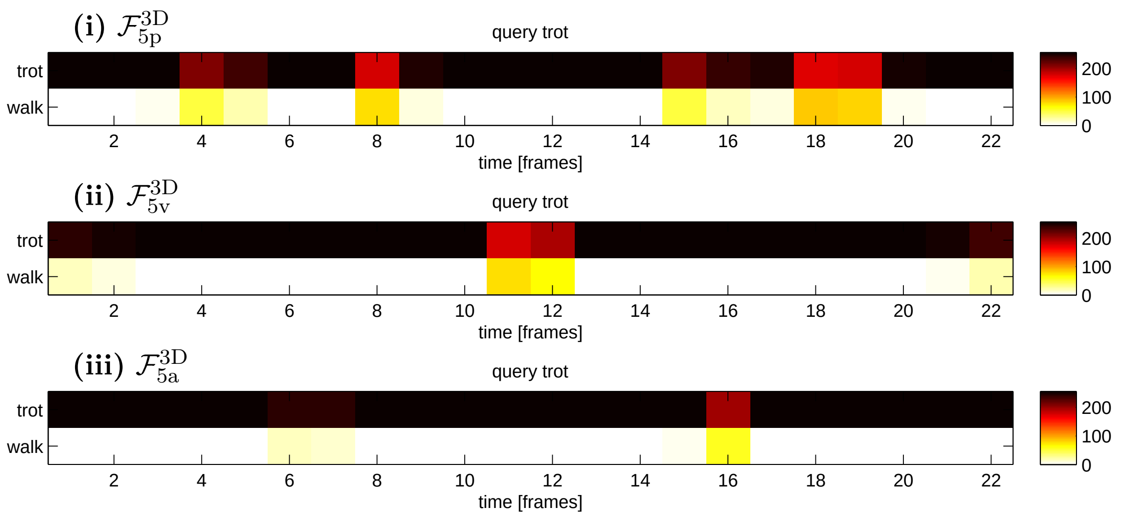
(b) K nearest neighbors retrieved through the LNG-based method.

Figure 8.3: Comparison between the Knn-based method and the LNG-based method on walking motion cycles: **(a)** illustrates the visualization of K nearest neighbors retrieved through the Knn-based retrieval approach. **(b)** explores the nearest neighbors retrieved according to the LNG paths per frame. We have conducted this experiment on all three feature sets developed on the basis of 3D information: **(i)** $\mathcal{F}^{3D}_{5p}$, **(ii)** $\mathcal{F}^{3D}_{5v}$ and **(iii)** $\mathcal{F}^{3D}_{5a}$.

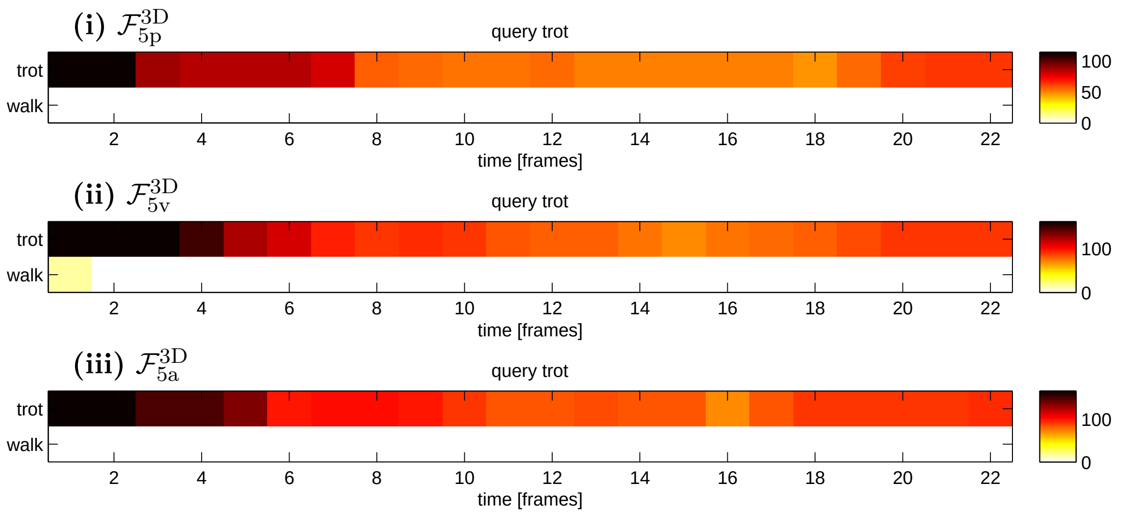
The third energy, E_s, imposes smoothness upon the current synthesized pose through exploiting the temporal information as,

$$E_s(\widetilde{\mathbf{X}}) = \frac{1}{\sqrt{\ell_t}} \cdot \sqrt{\|(\widetilde{\mathbf{X}}_t - 2\widehat{\mathbf{X}}_{t-1} + \widehat{\mathbf{X}}_{t-2})\|^2}. \tag{8.5}$$

The notations $\widehat{\mathbf{X}}_{t-1}$ and $\widehat{\mathbf{X}}_{t-2}$ are the reconstructed poses at frames $t-1$ and $t-2$ respectively.



(a) K nearest neighbors retrieved through the Knn-based method.



(b) K nearest neighbors retrieved through the LNG-based method.

Figure 8.4: Comparison between the Knn-based method and the LNG-based method on trotting motion cycles: **(a)** elaborates the visualization of K nearest neighbors retrieved through the Knn-based retrieval approach. **(b)** explores the nearest neighbors retrieved according to the LNG paths per frame. We conduct this experiment on all three feature sets developed on the basis of 3D information: **(i)** $\mathcal{F}^{3D}_{5p}$, **(ii)** $\mathcal{F}^{3D}_{5v}$ and **(iii)** $\mathcal{F}^{3D}_{5a}$.

8.5 Experimental Results

We evaluate the presented methods for retrieval and reconstruction on the MoCap data as well as on the video data. We design multiple feature sets for search and retrieval the similar poses from the MoCap dataset as mentioned in Section 8.3 and examine them in different ways in order to check their efficiencies. For video input based evaluations, we first capture the horse motions, where the horses perform same types of motions as perform for the MoCap data *e.g.* walking and trotting motions. We adjust the frame rate (*e.g.* 25 frames per second) for both, the MoCap dataset and the video input data. For performance metric with respect to the 3D motion reconstruction, we compute the *average reconstruction error* by calculating the average Euclidean distance in centimeters between the estimated 3D pose and the ground truth pose relative to the root joint (the marker *Withers* with the marker ID 9 in our marker set) per frame (Equation 4.13). We utilize all 15 joints of the skeleton as described in Table 8.1 for evaluation. We first conduct experiments to testify the search and retrieval methods and in the end we report on the effectiveness of our video-based motion reconstruction approach.

8.5.1 Similarity Searches

We evaluate the search and retrieval methods by conducting a series of experiments with different scenarios such as: (i) We first perform experiments in order to compare the simple Knn-based method with the proposed LNG-based variant on the motion capture data. (ii) We perform the numerical similarity searches to evaluate all three feature sets developed on the basis of 3D information like, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5p}}$, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5v}}$ and $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5a}}$. (iii) We also conduct experiment to report results for the logical similarity searches. (iv) In the end, we evaluate the proposed LNG-based variant on the video input data as well.

8.5.1.1 Knn-based and LNG-based Similarity Searches

In order to investigate the proposed LNG-based voting strategy against the simple Knn-based voting per frame, we perform experiments on the representative motion cycles for both motion classes, walking motions and trotting motions, and the results are shown in Figure 8.3 and Figure 8.4 respectively. The horizontal axis in these Figures describes a time line given in frames, while the vertical axis shows all retrieved similar motion classes from the MoCap dataset according to the query motion cycle. We search for 256 nearest neighbors and count motion classes per frame to which these nearest neighbors belong. This counted number of found nearest neighbors per frame is color coded from white (no neighbor found for this motion class) to black (256 neighbors found for this class). Consequently, these graphs show per frame confusion of the neighborhoods obtained with the respective methods. The similarity search has been performed for all three 3D feature sets, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5p}}$, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5v}}$ and $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5a}}$.

Knn-based method. For walking example, the most of the neighbors belong to the walking class. The only exception is frame 16 for the velocity based feature set. While for trot motion cycle, the more confusion between the two motion classes occurs but still the majority of the voting belongs to the correct motion class in all frames. The wrong voting results indicate that the Knn-based method is not stable enough even for this simple scenario where we take into account just only two motion classes. The more confusion for trotting motion cycle is due to the higher speed in that motion class as compared to the walking motion class. The results for the Knn-based method on both motion classes walking and trotting, have been shown in Figure 8.3(a) and Figure 8.4(a) respectively.

LNG-based method. In case of LNG-based method, Figure 8.3 (b) and Figure 8.4 (b) show results for walking and trotting motions respectively. We fix the window length ($\mathcal{Z} = 10$) for the graph construction in these examples. In contrast to the Knn-based method, for the LNG-based method we have found the similar results for both example motion classes. No mislabeling were found in both cases, in return the number of retrieved closest neighbors from the LNG paths are decreased after the first couple of frames. For the first few frames the results are the same as for the direct Knn-based voting due to the fact that at that point the path for LNG has a shorter length. Later on, with the full window length ($\mathcal{Z} = 10$), the path length increases and the number of nearest neighbors connected with the graph structure drops down.

8.5.1.2 Numerical Similarity Searches

We testify the developed 3D feature sets, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5p}}$, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5v}}$ and $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5a}}$ for the numerical similarity search. For that purpose, we search for the similar motion cycles using the graphical structure LNG. To come up with precision-recall diagrams, we extend the local pose neighborhood until all motion cycles of the query class are returned as match. We perform this experiment with all feature sets based on the 3D information and the results are reported in Figure 8.5, where the precision-recall diagrams has been drawn for walking and trotting motion cycles. From the results, we have found that for the walking query, we obtain a high precision value 97% up to a recall from 97%. For all feature sets, the precision drops for the last few matches only. In contrast to walking motion cycle, more mismatches are returned for the trot motion cycle when we use the position based feature set $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5p}}$. With the derived feature sets, $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5v}}$ and $\mathcal{F}^{\mathrm{3D}}_{\mathrm{5a}}$, we obtain much better results. This behavior can be elaborated by a closer look on the execution of these motion classes. In both motion classes, walk and trot, the marker positions are not sufficiently distinctive, while velocities and accelerations are very particular and distinctive for both types of motion classes due to the speed variations between these classes.

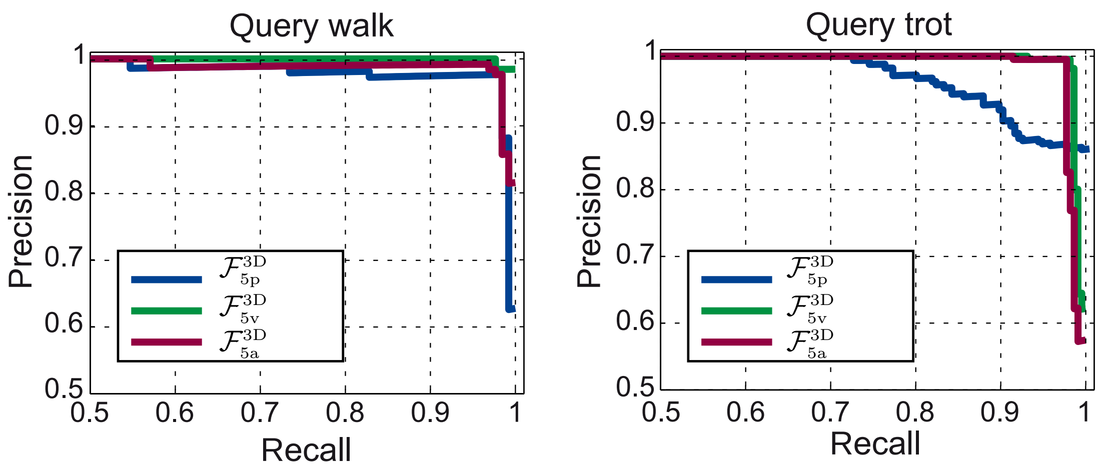


Figure 8.5: The precision-recall diagrams which show comparisons between the 3D feature sets developed on the basis of the 3D positions ($\mathcal{F}^{\mathrm{3D}}_{\mathrm{5p}}$), velocities ($\mathcal{F}^{\mathrm{3D}}_{\mathrm{5v}}$) and accelerations ($\mathcal{F}^{\mathrm{3D}}_{\mathrm{5a}}$). We show results for one representative query motion cycle of both motion classes: walking (**left**) and trotting (**right**).

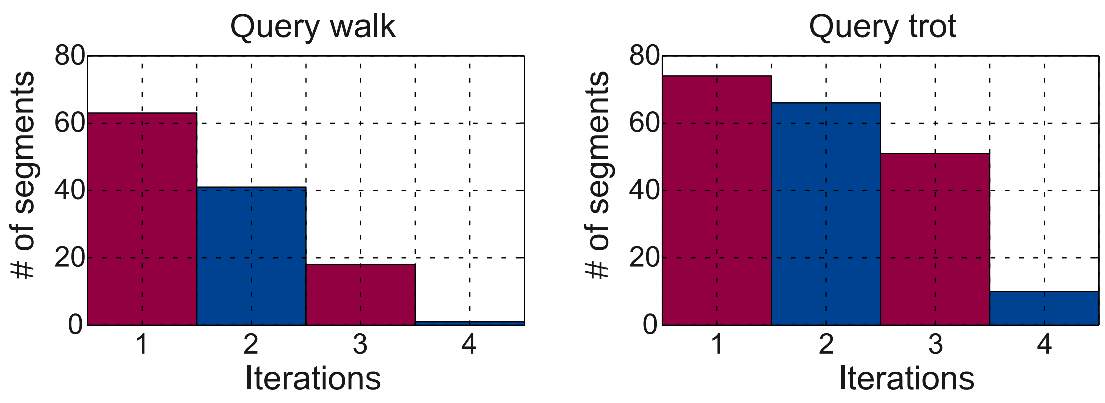


Figure 8.6: The number of motion segments found per iteration for the logical similarity search. We show results for one representative query motion cycle of the two motion classes: walking (**left**) and trotting (**right**).

8.5.1.3 Logical Similarity Searches

Kovar and Gleicher [KG04] introduced the concept of logical similarity searches, where the retrieved matches of a query motion segment are used as new queries in new iteration of the searching process. The process continues and the new segments are retrieved until no any new segment is found. We perform this experiment with the motion cycles from both classes. To this end, we restrict the number of nearest neighbors to 256 in order to ensure that no false positives are returned for a query motion cycle. In both cases, walk and trot, this retrieval scenario finds roughly all motion cycles without any mismatch. Figure 8.6 shows the numbers of the new found motion cycles per iteration. The algorithm converges after four iterations in both cases.

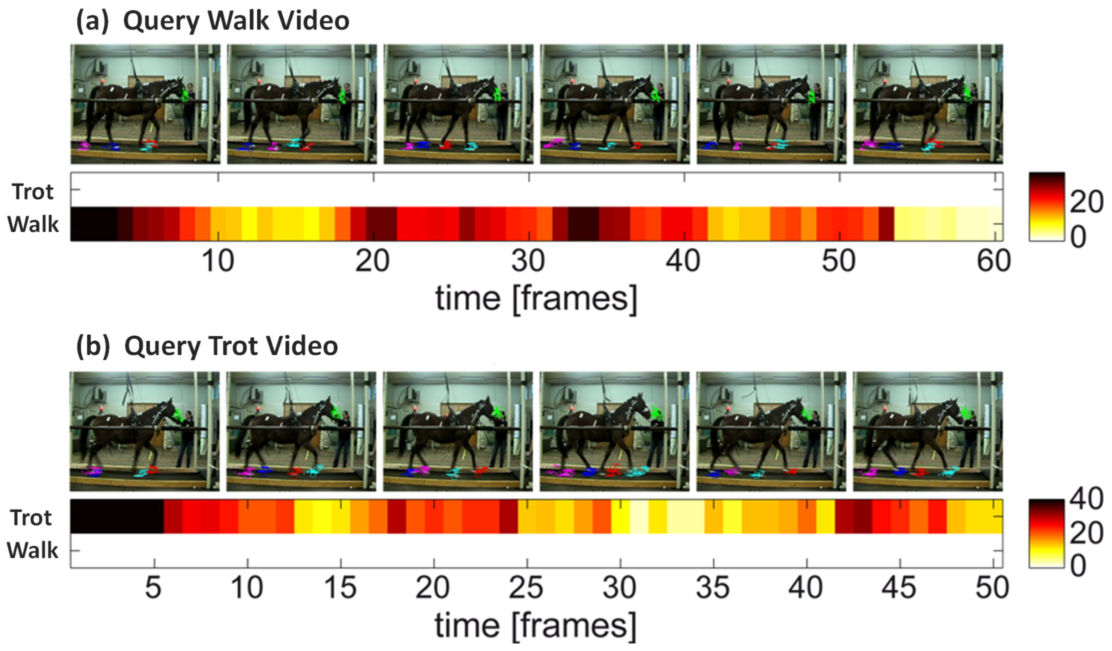


Figure 8.7: The number of the LNG paths per frame for walking and trotting motion cycles. The K nearest neighbors are searched into the MoCap dataset on the basis of the 2D feature sets $\mathcal{F}_5^{\mathrm{vid}}$ detected and tracked in the input video sequences.

8.5.1.4 Video Input Data

As we also propose a methodology of the 3D reconstruction of the horse motion from input video data, in this context we test LNG-based method on video input motion sequences as well. In this case, we track and extract the feature set $\mathcal{F}_5^{\mathrm{vid}}$ for each video motion class. We perform Knn search for 256 nearest neighbors into the MoCap dataset $\mathcal{DB}_{\mathrm{quad}}^{25\mathrm{Hz}}$ and compute the LNG paths accordingly. We present the results as well as a few example frames of the input video sequence with projected nearest neighbors in Figure 8.7. We retrieve nearest neighbors with correct motion classes for all considered frames. There are a few frames, especially in the trot example, where a very low number of paths are returned. Nevertheless all of these paths belong to the correct motion class and we get very good nearest neighbors when we employ the LNG-based retrieval methodology.

Due to the small MoCap dataset, the K nearest neighbors are retrieved relatively fast. The construction of the kd-tree on the database $\mathcal{DB}_{\mathrm{quad}}^{30\mathrm{Hz}}$ is done in less than 0.8 milliseconds and the searching for $\mathsf{K} = 256$ similar poses takes 0.6 milliseconds on an average.

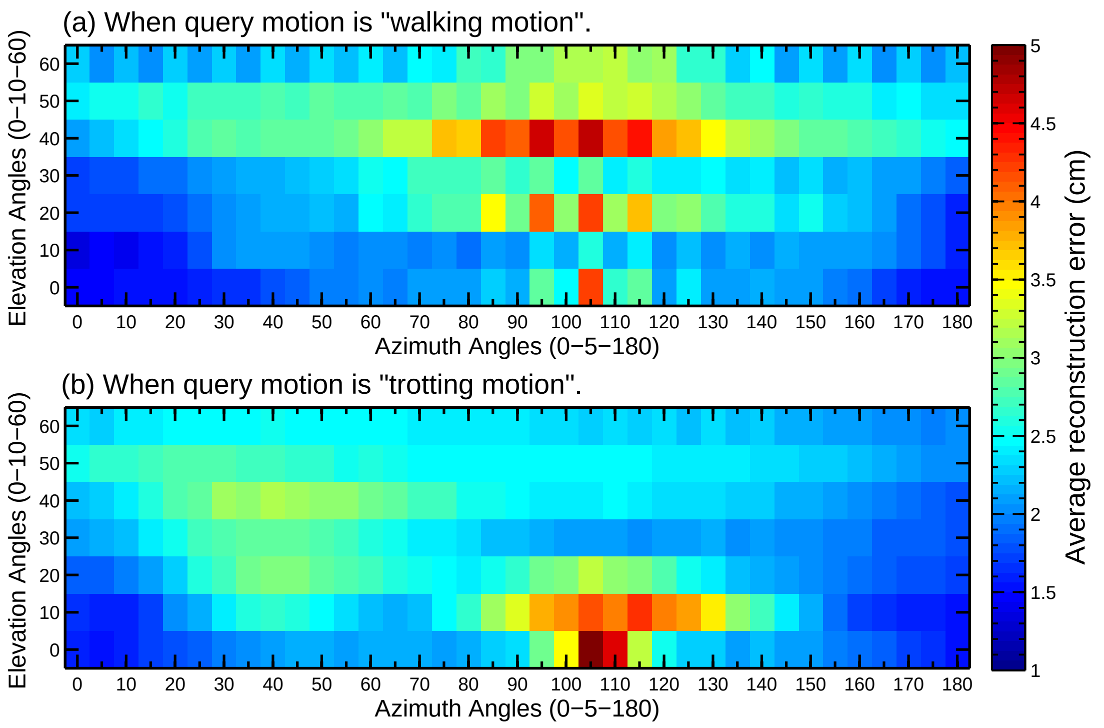


Figure 8.8: The average reconstruction errors for walking and trotting motions at different test view directions—the azimuth angles (0–5–180) degrees with step size 5 degree and the elevation angles (0–10–60) degrees with step size 10 degree.

8.5.2 Video based Reconstructions

We evaluate the performance of the proposed reconstruction methodology quantitatively as well as qualitatively on two types of input examples *e.g.* synthetic examples obtained from the MoCap files by some random camera parameters and the video examples.

8.5.2.1 Synthetic Inputs

In case of synthetic input examples, we test our approach on both types of motion sequences *e.g.*, walking and trotting motion sequences at wide range of test view directions—the azimuth angles (0–5–180) degrees with 5 degree step size, assuming that the same results would be executed for other half of the circle and the elevation angles (0–10–60) degrees. In case of the elevation angles, we consider the fact that for the head-mounted cameras near to 90 degree or *top view*, the body of the performing horse becomes an hindrance in capturing the full detailed motions of the hooves, that's why we fix the range of the elevation angles from 0 degree to 60 degree with 10 degree step size. The initial view pose is the right side of the performing horse and it is the starting point for all view directions with azimuth and elevation angles.

We report the results in Figure 8.8, where the average reconstruction errors for different test views are shown in the form of graph with the azimuth angles (0–5–180) along x-axis and the elevation angles (0–10–60) along y-axis, while the reconstruction error is color-coded from *blue* (low error) to *red* (high error).

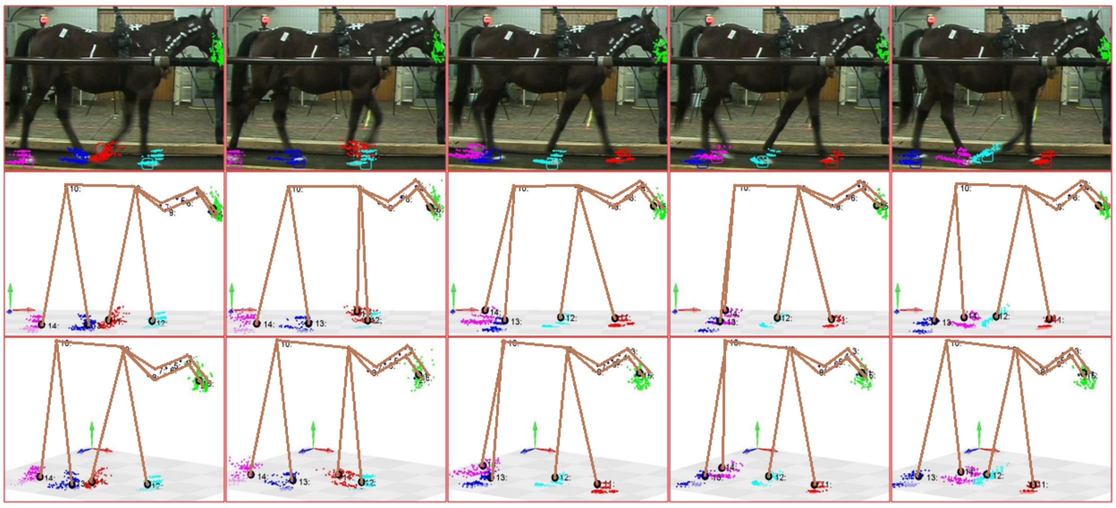
Walking Motions. For walking motion sequences, it is observed that at the *side view* either it is left side or right side, the lowest reconstruction errors are executed because of the fact that the movements of the hooves of the performing horse are much more elaborate and can be viewed and captured in detail. In *front view*, due to lack of details in motion information, the reconstruction errors increase comparatively. It is also observed that for the view directions near to the head-mounted camera view (*top view*), the reconstruction error also increases as expected which is quite evident from Figure 8.8(a).

Trotting Motion. We observe the similar behaviour for trotting motions as reported in Figure 8.8(b). Like walking motion, for *side view*, the reconstruction error is the lowest. The roughly identical results to the walking motion, have been discovered in case of the *front view* and the *top view*. The little bit difference between the results for both motion classes is due to that for trotting motions, the movement patterns for the positions of the end effectors (the hooves and the head) are a bit different from the walking motion sequences (see Section 8.2.2).

8.5.2.2 Video Inputs

We check our reconstruction approach qualitatively on real videos of walking and trotting motions. The 2D feature sets $\mathcal{F}_5^{\mathrm{vid}}$ extracted from the input video are given as input to the system. On the basis of Knn retrieved through the 2D feature sets, we reconstruct the final 3D horse motions. A few qualitative reconstruction results for the video based motions are shown in Figure 8.9. For real video of the horse motion, sometime, we cannot detect and track the 2D feature sets correctly due to blurring effects, occlusion and illuminations *etc.* especially in case of trotting motions, and consequently it may impact on the final reconstruction process. To avoid this situation, we have annotated the key-frames of the video manually. We have found from the experiments that the proposed system executes acceptable results even with a noisy input data.

(a) When the walking motion sequences are given as input query.



(b) When the trotting motion sequences are given as input query.

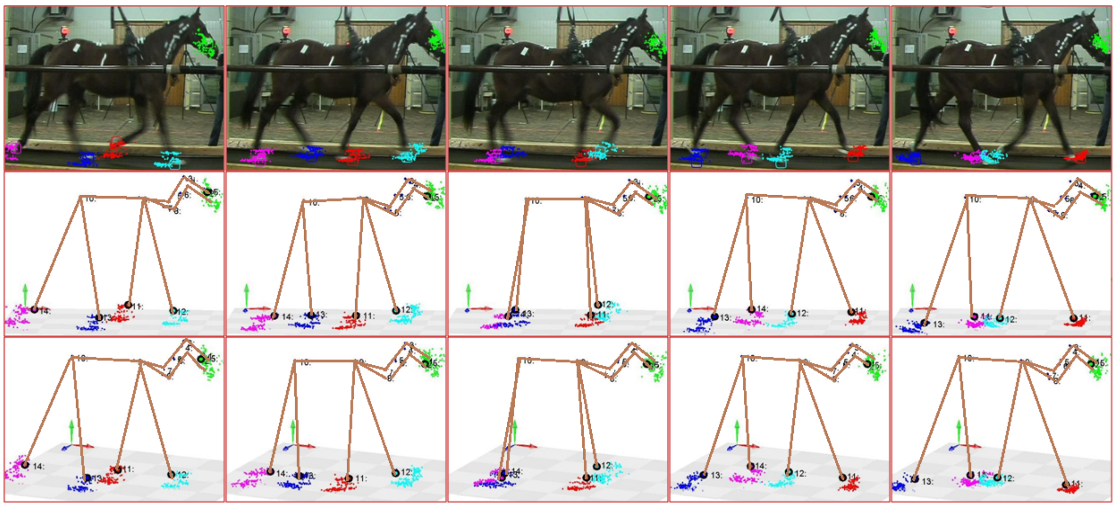


Figure 8.9: A few qualitative results for the 3D reconstructions, when the monocular video sequences for walking and trotting motions are given as input query. **(a)** and **(b)** represent a few frames of walking and trotting motions respectively. The **(first rows)** show video input frames with detected 2D feature sets and the projected Knn. The **(second rows)** represent the corresponding 3D reconstructions with Knn, while the **(third rows)** correspond to the 3D reconstructions at other viewpoint.

8.6 Conclusion

In this chapter, we have transferred techniques developed for the human motion data to the motion data from the quadrupeds. For motion retrieval, we identify and develop suitable feature sets based on the 3D as well as 2D information and utilize them to retrieve nearest neighbors from the MoCap dataset. We evaluate the proposed variant of the LNG-based method against the simple Knn-based retrieval method and have found that the LNG-based method retrieve very good nearest neighbors comparatively. For motion reconstruction from video data, even having lack of information like skeleton data, joint angle configurations and with a few markers, we obtain very satisfying results in both cases, for the synthetic 2D input query motions as well as for the real video sequences. The presented online reconstruction framework reconstructs 3D motions with approximately 6–8 frames per second.

9 Conclusion and Future Perspectives

To know, is to know that you know nothing. That is the meaning of true knowledge.

Socrates

In this dissertation, we have addressed the problem of 3D human pose estimation from a monocular video or from a single RGB image. In order to cope this, we have proposed a few pipelines and algorithms for retrieval and reconstruction of the 3D poses from different types of input control signals like 2D synthetic video/image or real video/image that have been captured in indoor studio-like controlled environments or outdoor uncontrolled natural environments. The main focus of the dissertation is to reconstruct 3D human poses without relying on any special setups *i.e.* retro-reflective markers, Inertial Measurement Units, depth camera or multi-camera system. Moreover, we do not take into consideration any image cues like depth information, background subtraction or bounding boxes *etc.* The proposed systems are flexible enough and can reconstruct 3D poses in any indoor/outdoor environment.

The main advantage of our proposed methodologies is, in order to infer final 3D human pose we do not need any synchronized 3D-2D pose-image pairs which is not only costly but also problematic with respect to creating such a studio-like setup and camera calibrations. We integrate both media, the 3D marker-based motion capturing and a simple RGB camera based capturing of photographs or motions, together for 3D pose estimation and as a result we fill the gap between these separate media. In the end, we also extend the proposed retrieval and reconstruction methods to quadruped pose estimation.

9.1 Video-based Retrieval and Reconstruction

In first part of the dissertation, we have proposed framework for robust retrieval and reconstruction of human motions from a video data. We first detect feature sets using SURF, MSER and colorMSER local feature detectors/descriptors and track them through matching by developing a dictionary of features (DOFs). We have resolved the 3D-2D cross models retrieval problem by developing the *knowledge base*, which contains 3D normalized poses as well as the corresponding 2D normalized poses. We develop 2D normalized pose space through sampling of the MoCap dataset at different view directions. Using these virtual cameras, we create correspondence and synchronization between 3D-2D poses and as a result we retrieve the closest 3D motion segments from the MoCap dataset efficiently by employing just a subset of 2D joints. We construct a local 3D pose model in low dimensional pose space from these retrieved 3D similar poses, which is optimized through multiple energy terms. One of the main strengths of our proposed pipeline is, we utilize just only five joints such as the four end effectors and the head to perform Knn search and to reconstruct final 3D poses. Utilizing these five joints, we even handle the motions that are recorded particularly through head mounted cameras with top view which often capture human postures with ambiguities and indistinctive 2D joint positions. We have thoroughly evaluated our approach with a variety of experimental setups which are designed in terms of performing actors, view directions and the noisy inputs. Our approach achieves very good results for all these different experiments.

The robust and optimal nearest neighbors have a great influence in local modeling and the final 3D reconstruction. In this context, we then make use of temporal coherence of the control input signals in searching and retrieval of the closest examples by constructing a graphical structure *online lazy neighborhood graph*. With OLNG we have improved not only the search and retrieval process but also the ultimate reconstruction methodology. We formulate the symmetric square root kernel function in order to estimate the probability density of the MoCap priors for final 3D motion reconstruction from video input sequences. We also get benefits from the 3D MoCap priors and make the image-based feature detection and tracking more accurate and robust. To accomplish this, we project the retrieved 3D poses to the image plane by estimated camera parameters and then weight for the more accurate features. With the use of priors, we have increased the tracking rate up to 15-20% by handling occlusion and the blurring effects. Our proposed systems need a bit pre-processing and can reconstruct motions in a realtime.

9.2 Image-based Retrieval and Reconstruction

In second part of the dissertation, we address the most challenging task to estimate the 3D human pose from a single monocular image which is considered as severely underconstrained problem. We propose methodology to recover the 3D human pose, when a single monocular image, either synthetic or real RGB image, is given as input to the developed system. With the given 2D joint positions, we search into the MoCap dataset for similar poses. As we work on the sparse input control signals without considering any temporal coherence, the searching and retrieval of Knn from MoCap dataset is not too elaborate. In this context, we design and devise multiple feature sets composed of on subsets of skeleton joints and then make comparisons between these feature sets with respect to accuracy, time and memory. We also introduce two-fold method to estimate the camera parameters where we benefit from the virtual cameras through which we have performed sampling of the MoCap dataset as well as from the retrieved K closest examples which we map onto 2D estimated feature sets in order to minimize the projection error. We compare our approach against state-of-the-art methods and have found that our approach outperforms all state-of-the-arts. Our approach produces state-of-the-art results even in case of noisy 2D inputs, which are often resulted in capturing photographs in outdoor natural environments. Moreover, the presented system achieves competitive results when the test data and the MoCap data are from different datasets. We also evaluate our approach qualitatively on the real internet images as well as on the hand-drawn sketches and we have found from the results that our approach yields very plausible 3D poses even for erroneous and ambiguous input 2D poses.

We propose a dual source approach where we estimate 3D pose by deploying two training sources, images with annotated 2D poses and the 3D MoCap data. We integrate both sources together and introduce a dual-source methodology in order to infer the final 3D human pose. This makes the approach very practical since annotating images with accurate 3D poses is often infeasible while 2D pose annotations of images and motion capture data can be collected separately without much effort. We first learn a regressor for 2D pose estimation from the image source data and is given as input for Knn search and retrieval. We introduce efficient searching approach that is robust to 2D pose estimation errors, where we collectively use multiple feature sets developed on the basis of different subsets of body joints. Moreover, our proposed system also refines the initial estimated 2D input pose and can be iterative to attain more accurate 3D pose. Our proposed approach achieves state-of-the-art results when the training data are from the same dataset. Even with other training dataset, our approach produces acceptable results which are comparable to other state-of-the-arts too.

9.3 Quadruped Motions

In the last part of the dissertation, we have presented the retrieval and reconstruction from quadruped motions. We have adapted the human motion retrieval and reconstruction techniques to work with quadruped motions. In this way, we bridge the gap between the computer animation techniques specific to the human motions and the quadruped motions. Considering the results from different experiments, we have shown that extending the Knn search by a temporal component, even for a simple dataset can lead to good results. For video-based 3D motion reconstruction, the presented system produces satisfying results in approximately realtime utilizing just a few markers (markers of the hooves and the head) without having skeleton parametrization as well as the joint angle configurations.

9.4 Future Perspectives

Finally, the work at hand opens several other interesting directions and perspectives for future work such as:

At the moment, we are dealing with a static monocular camera which can be replaced by multiple moving cameras capturing poses in indoor/outdoor environments. Using multi-camera system, it may provide more cues regarding person's posture in the scene using epipolar geometry. In case of video-based feature extraction, other image cues like edges, silhouette or depth can be combined with SURF and MSER feature detectors to make detection more robust and extend the proposed methodology to a wider range of motion classes and actions. In addition, the estimation of the full perspective camera model with 11 degrees of freedom can be another aspect of the future work.

In this dissertation, we work on estimation of the 3D human poses for a single person that presents in a video or in an image. This work can be extended to multiple persons interacting with each other or with different objects *e.g.* table, chair or working in office/kitchen *etc.* simultaneously. In this context, the extension will produce flexibility and bring the nature more closer. Furthermore, the proposed method for human pose estimation can be coupled with gait analysis and the person identifications. As a result, the system's usability and applicability may increase significantly. One might integrate action recognition and pose estimation together in 3D-3D, 2D-3D and 3D-2D scenarios. The cues specific to some approach might be helpful in other approach as well.

Another important dimension would be the use of multiple setups and the hardware such as depth cameras or Inertial Measurement Units integrated with video sequences. The proposed video-based reconstruction method for full body pose estimation can be leveraged with the depth information or acceleration sensors. These sensors have been used to reconstruct human motions [TZK+11].

The reconstruction methodologies can be improved further by adding physics-based properties *i.e.* velocity, acceleration, body weight and mass, force of gravity, angular momentum, external forces *etc.* The physics-based constrains may yield the more natural looking 3D poses which is basically the ultimate target of any application with respect to people analysis and the understanding of their motions. The retrieval and reconstruction on the basis of hand-drawn sketches can be performed on more detailed input scenarios with a variety of input action classes. Additionally, the anthropometric property and bone-length constraints can be imposed on the reconstruction approach which may produce more plausible 3D poses.

For quadruped motions, one of the most important directions for future work is the creation of an enlarged motion capture database with more types of gait and other typical exercises. The skeleton representation of the quadruped might be computed and helpful in the process of full body quadruped motion retrieval and reconstruction. With such type of data at hand, more sophisticated techniques for retrieval and reconstruction might be applied and compared in a more reasonable manner. We have presented results on a static camera with an object that is moving on a treadmill only. The quadruped motion reconstruction might be extended to more complex scenarios like reconstruction of the motion sequences from a riding theater captured through a single camera or multi-camera system which is one of the crucial future directions. In this scenario, the types of motions are less restricted as compared to the current treadmill scenario. Another possible strand of research is the reconstruction of full-body movements based on accelerometer readings combined with the video based cues in outdoor scenarios. One might record other quadruped species as well in order to derive more general model of quadruped motions for such kind of data.

Bibliography

[AB15] Ijaz Akhter and Michael J. Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. xii, 5, 17, 23, 27, 29, 42, 67

[ARS10] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Monocular 3d pose estimation and tracking by detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 26, 26

[AT04] Ankur Agarwal and Bill Triggs. 3d human pose from silhouettes by relevance vector regression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 23, 25, 25

[AT06a] Ankur Agarwal and Bill Triggs. A local basis representation for estimating human pose from cluttered images. In *Asian Conference on Computer Vision (ACCV)*, 2006. 17, 32

[AT06b] Ankur Agarwal and Bill Triggs. Recovering 3d human pose from monocular images. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006. 23, 32, 104

[BAA+14] Vasileios Belagiannis, Sikandar Amin, Mykhaylo Andriluka, Bernt Schiele, Nassir Navab, and Slobodan Ilic. 3d pictorial structures for multiple human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 27

[BETVG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up Robust Features (SURF). *Comput. Vis. Image Underst.*, 110(3), 2008. 19, 20, 31, 36, 42

[BM98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998. 16, 16

[BMB+11] Andreas Baak, Meinard Muller, Gaurav Bharaj, Hans-Peter Seidel, and Christian Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *International Conference on Computer Vision (ICCV)*, 2011. 28

[BMP04] Christoph Bregler, Jitendra Malik, and Katherine Pullen. Twist based acquisition and tracking of animal and human kinematics. *Int. J. Comput. Vision*, 56(3), 2004. 32

[BOS00] H H Buchner, S Obermüller, and M Scheidl. Body centre of mass movement in the sound horse. *Veterinary journal (London, England)*, 160(3), 2000. 129

[BS10] Liefeng Bo and Cristian Sminchisescu. Twin gaussian processes for structured prediction. *International Journal of Computer Vision (IJCV)*, 2010. 23, 32, 104, 114, 114, 114, 114, 114, 114, 115, 115, 115, 115, 116, 117, 117, 117, 119, 120

[BSB+07] A. O. Bălan, L. Sigal, M. J. Black, J. E. Davis, and H. W. Haussecker. Detailed human shape and pose from images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 29, 45

[BSKM08] Liefeng Bo, Cristian Sminchisescu, Atul Kanaujia, and Dimitris Metaxas. Fast algorithms for large scale conditional 3d prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 23, 32, 104

[CC09] Yen-Lin Chen and Jinxiang Chai. 3d reconstruction of human motion and skeleton from uncalibrated monocular video. In *Asian Conference on Computer Vision (ACCV)*, 2009. 23, 25, 26, 26

[CCH12] Worawat Choensawat, Woong Choi, and Kozaburo Hachimura. Similarity retrieval of motion capture data based on derivative features. *JACIII*, 16(1), 2012. 33

[CG11] Aaron Chavez and David Gustafson. Color-based extensions to msers. In *ISVC (2)*, Lecture Notes in Computer Science, 2011. 20, 31

[CH05] Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. *ACM Transactions on Graphics (ToG)*, 24(3), 2005. 25, 26, 27, 29, 32, 37, 37, 37, 37, 37, 45, 45

[CMU14] CMU. Carnegie mellon university graphics lab: Motion capture database, 2014. `mocap.cs.cmu.edu`. 3, 5, 33, 81, 90, 104, 111, 117, 129

[CYI+12] M. G. Choi, K. Yang, T. Igarashi, J. Mitani, and J. Lee. Retrieval and visualization of human motion data via stick figures. *Comput. Graph. Forum*, 31(7pt1), 2012. 30, 31

[DAC+03] James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popović, and David Salesin. A sketching interface for articulated figure animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2003. 30, 30

[DB06] Michael Donoser and Horst Bischof. Efficient maximally stable extremal region (mser) tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 36, 42

[DBR00] Jonathan Deutscher, Andrew Blake, and Ian D. Reid. Articulated body motion capture by annealed particle filtering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000. 23

[DLGVG14] Matthias Dantone, Christian Leistner, Juergen Gall, and Luc Van Gool. Body parts dependent joint regressors for human pose estimation in still images. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014. 21, 21, 21, 21, 31, 32, 106, 107, 107, 111, 116, 118

[DR05] Jonathan Deutscher and Ian Reid. Articulated body motion capture by stochastic search. *Int. J. Comput. Vision*, 61(2), 2005. 18, 31, 32

[DR12] Chaitanya Desai and Deva Ramanan. Detecting actions, poses, and objects with relational phraselets. In *European Conference on Computer Vision (ECCV)*, 2012. 116, 118

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 19, 31

[EsL04] Ahmed Elgammal and Chan su Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 25

[FB02] Ronan Fablet and Michael J. Black. Automatic detection and tracking of human motion with a view-based representation. In *European Conference on Computer Vision-Part (ECCV)*, 2002. 32

[FE73] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Comput.*, 22(1), 1973. 20

[FH05] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1), 2005. 21, 31

[FMR08] Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 21, 31

[FRDC04] Laurent Favreau, Lionel Reveret, Christine Depraz, and Marie-Paule Cani. Animal gaits from video. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2004. 21, 21, 22, 33

[FZZW14] Xiaochuan Fan, Kang Zheng, Youjie Zhou, and Song Wang. Pose locality constrained representation for 3d human pose reconstruction. In *European Conference on Computer Vision (ECCV)*, 2014. 17, 27, 29, 29, 30, 83, 86, 89, 89, 90, 90, 91, 91, 91, 96, 96, 97, 98, 98, 102

[GFB12] Gutemberg Guerra-Filho and Arnab Biswas. The human motion database: A cognitive and parametric sampling of human motion. *Image and Vision Computing*, 2012. 3, 33

[GWK05] Daniel Grest, Jan Woetzel, and Reinhard Koch. Nonlinear body pose estimation from depth images. In *27th DAGM Conference on Pattern Recognition*, 2005. xii, 16, 17, 28

[HDK07] Alexander Hornung, Ellen Dekkers, and Leif Kobbelt. Character animation from 2d pictures and 3d motion data. *ACM Transactions on Graphics (ToG)*, 26(1), 2007. 27, 29

[HHL13] Ting-Chieh Huang, Yi-Jheng Huang, and Wen-Chieh Lin. Real-time horse gait synthesis. *Computer Animation and Virtual Worlds (CAVW)*, 2013. 21, 21, 21, 33, 129

[HLR$^+$10] S J Hobbs, D Levine, J Richards, H Clayton, J Tate, and R Walker. Motion analysis and its use in equine practice and research. *Wiener Tierärztliche Monatsschrift*, 97, 2010. 125

[ICS14] C. Ionescu, Joao Carreira, and C. Sminchisescu. Iterated Second-Order Label Sensitive Pooling for 3D Human Pose Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 23, 104

[IPOS14] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014. xviii, 7, 33, 104, 104, 111, 118, 121, 123

[JE10] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *British Machine Vision Conference (BMVC)*, 2010. xviii, 7, 122, 123

[JSMH12] Eakta Jain, Yaser Sheikh, Moshe Mahler, and Jessica Hodgins. Three-dimensional proxies for hand-drawn characters. *ACM Transactions on Graphics (ToG)*, 31(1), 2012. 29, 30, 30, 45

[KB01] Timor Kadir and Michael Brady. Saliency, scale and image description. *Int. J. Comput. Vision*, 45(2), 2001. 19, 31

[KG04] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics (ToG)*, 23(3), 2004. 32, 137

[KG14] Ilya Kostrikov and Juergen Gall. Depth sweep regression forests for estimating 3d human pose from images. In *British Machine Vision Conference (BMVC)*, 2014. 24, 24, 32, 104, 111, 111, 114, 114, 115, 115, 116, 116, 119, 119, 119, 120

[KSM07] Atul Kanaujia, Cristian Sminchisescu, and Dimitris N. Metaxas. Semi-supervised hierarchical models for 3d human pose reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 32

[KTWZ10] Björn Krüger, Jochen Tautges, Andreas Weber, and Arno Zinke. Fast local and global similarity searches in large motion capture databases. In *ACM SIGGRAPH Symposium on Computer Animation (SCA)*, 2010. 32, 33, 37, 39, 39, 84, 84, 94, 108, 129

[LC14] Sijin Li and Antoni B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision (ACCV)*, 2014. 16, 17, 23, 27, 30

[LCR$^+$02] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics (ToG)*, 21(3), 2002. 25

[Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2), 2004. 18, 31

[LZC15] Sijin Li, Weichen Zhang, and Antoni Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *ICCV*, 2015. 23

[MA06] David M. Mount and Sunil Arya. ANN: a library for approximate nearest neighbor searching. Programming manual, Department of Computer Science, University of Maryland, College Park, Maryland, U.S.A., 2006. 45

[MCUP02] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference (BMVC)*, 2002. 20, 31, 36, 42

[Men10] Alberto Menache. *Understanding Motion Capture for Computer Animation, Second Edition.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2010. 1, 2

[MHK06] T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding (CVIU)*, 104(2), 2006. 125

[MM06] Greg Mori and Jitendra Malik. Recovering 3d human body configurations using shape contexts. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006. 23

[MQW07] Chen Mao, Sheng Feng Qin, and David Wright. Sketch-based virtual human modelling and animation. In *8th International Symposium on Smart Graphics*, 2007. 30

[MRC05] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. *ACM Transactions on Graphics (ToG)*, 24, 2005. 32, 33

[MRC$^+$07] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, 2007. 3, 6, 33, 48, 72, 81, 90, 129

[MSZ94] Richard M. Murray, S. Shankar Sastry, and Li Zexiang. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1994. 14, 15

[NKMC09] Sandra Nauwelaerts, Leeann Kaiser, Robert Malinowski, and Hilary M Clayton. Effects of trunk deformation on trunk center of mass mechanical energy estimates in the moving horse, equus caballus. *Journal of Biomechanics*, 42(3), 2009. 129

[OS08] R. Okada and S. Soatto. Relevant feature selection for human pose estimation and localization in cluttered images. In *European Conference on Computer Vision (ECCV)*, 2008. 32

[PCS02] Min Je Park, Min Gyu Choi, and Sung Yong Shin. Human motion reconstruction from inter-frame feature correspondences of a single video stream using a motion library. In *ACM SIGGRAPH/Eurographics symposium on Computer Animation (SCA)*, 2002. 25, 26, 26

[PMBG$^+$11] Gerard Pons-Moll, Andreas Baak, Juergen Gall, Laura Leal-Taixe, Meinard Mueller, Hans-Peter Seidel, and Bodo Rosenhahn. Outdoor human motion capture using inverse kinematics and von mises-fisher sampling. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 23

[PMFR14] Gerard Pons-Moll, David J. Fleet, and Bodo Rosenhahn. Posebits for monocular human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 4, 16, 24, 29, 30, 42, 67

[PS11] Hyun Soo Park and Yaser Sheikh. 3d reconstruction of a smooth articulated trajectory from a monocular image sequence. In *International Conference on Computer Vision (ICCV)*, 2011. 26

[Ram07] Deva Ramanan. Learning to parse images of articulated bodies. In *Advances in Neural Information Processing Systems (NIPS)*. 2007. xvii, 7, 81, 91, 101, 102

[RB08] Nadiya Roodsarabi and Alireza Behrad. 3d human motion reconstruction using video processing. In *Image and Signal Processing*, Lecture Notes in Computer Science. 2008. 26

[RDG13] Ibrahim Radwan, Abhinav Dhall, and Roland Goecke. Monocular image 3d human pose estimation under self-occlusion. In *The IEEE International Conference on Computer Vision (ICCV)*, 2013. 27, 32, 114, 114, 115, 115

[RKS12] Varun Ramakrishna, Takeo Kanade, and Yaser Ajmal Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European Conference on Computer Vision (ECCV)*, 2012. 17, 27, 29, 29, 29, 30, 30, 83, 86, 89, 90, 91, 96, 96, 97, 97, 98, 102

[RSB04] Stefan Roth, Leonid Sigal, and Michael J. Black. Gibbs likelihoods for bayesian tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 16, 16

[RSB+08] Bodo Rosenhahn, Christian Schmaltz, Thomas Brox, Joachim Weickert, and Hans-Peter Seidel. Staying well grounded in markerless motion capture. In *30th DAGM symposium on Pattern Recognition*, pages 385–395. Springer-Verlag, 2008. 26, 26, 27

[RSH+05] Liu Ren, Gregory Shakhnarovich, Jessica K. Hodgins, Hanspeter Pfister, and Paul Viola. Learning silhouette features for control of human motion. *ACM Transactions on Graphics (ToG)*, 24(4), 2005. 18, 25, 25, 25, 26

[SB06a] L. Sigal and M. J. Black. Hierarchical approach for articulated 3D pose-estimation and tracking. In *Learning, Representation and Context for Human Sensing in Video Workshop*, 2006. 23, 24

[SB06b] L. Sigal and M. J. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. xii, 16, 16, 16, 17

[SBB07] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. Combined discriminative and generative articulated pose and non-rigid shape estimation. In *Advances in Neural Information Processing Systems (NIPS)*, 2007. xii, xii, 16, 16, 17

[SBB10] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision (IJCV)*, 2010. 7, 33, 104, 111

[SBF00] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conference on Computer Vision-Part (ECCV)*, 2000. xii, 16, 16, 16, 17

[SFC+11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 28

[SH12] Hubert Shum and Edmond S.L. Ho. Real-time physical modelling of character movements with microsoft kinect. In *18th ACM Symposium on Virtual Reality Software and Technology (VRST)*, 2012. 31, 31

[Sho85] Ken Shoemake. Animating rotation with quaternion curves. *ACM Transactions on Graphics (ToG)*, 19(3), 1985. 14, 15

[SHP04] Alla Safonova, Jessica K. Hodgins, and Nancy S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics (ToG)*, 2004. 29, 45

[SIHB12] Leonid Sigal, Michael Isard, Horst Haussecker, and Michael J Black. Loose-limbed people: Estimating 3d human pose and motion using non-parametric belief propagation. *International Journal of Computer Vision (IJCV)*, 2012. 27

[SKLM05] Cristian Sminchisescu, Atul Kanaujia, Zhiguo Li, and Dimitris N. Metaxas. Discriminative density propagation for 3d human motion estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 23, 25, 32

[SLK11] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 32

[SMH06] Jonathan Starck, Gregor Miller, and Adrian Hilton. Volumetric stereo with silhouette and feature constraints. In *British Machine Vision Conference (BMVC)*, 2006. 18, 32

[SRH$^+$08] Ljiljana Skrba, Lionel Reveret, Franck Hetroy, Marie-Paule Cani, and Carol O'Sullivan. Quadruped animation. In *Eurographics State-of-the-Art Report (EG-STAR)*, 2008. 33

[SSQTMN13] Edgar Simo-Serra, Ariadna Quattoni, Carme Torras, and Francesc Moreno-Noguer. A Joint Model for 2D and 3D Pose Estimation from a Single Image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 16, 17, 24, 24, 27, 29, 29, 30, 30, 30, 30, 111, 111, 114, 114, 115, 115

[SSRA$^+$12] Edgar Simo-Serra, Arnau Ramisa, Guillem Alenyà, Carme Torras, and Francesc Moreno-Noguer. Single Image 3D Human Pose Estimation from Noisy Observations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 16, 17, 27, 29, 29, 29, 30, 30, 32, 111, 114, 114, 115, 115

[ST01] Cristian Sminchisescu and Bill Triggs. Covariance scaled sampling for monocular 3d body tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 16, 16

[ST02] Cristian Sminchisescu and Alexandru Telea. Human pose estimation from silhouettes. A consistent approach using distance level sets. *J. WSCG*,

2002. Special Issue: International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, (WSCG). 25, 25

[ST03a] Cristian Sminchisescu and Bill Triggs. Estimating articulated human motion with covariance scaled sampling. *I. J. Robotic Res.*, 22(6), 2003. 18, 31, 32, 32

[ST03b] Cristian Sminchisescu and Bill Triggs. Kinematic jump processes for monocular 3d human tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. 17

[SU10a] Mathieu Salzmann and Raquel Urtasun. Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 24

[SU10b] Mathieu Salzmann and Raquel Urtasun. Implicitly constrained gaussian process regression for monocular non-rigid pose estimation. In *Advances in Neural Information Processing Systems (NIPS)*. 2010. 32

[SVD03] Gregory Shakhnarovich, Paul Viola, and Trevor Darrell. Fast pose estimation with parameter-sensitive hashing. In *IEEE International Conference on Computer Vision (ICCV)*, 2003. 32

[TZK$^+$11] Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Transactions on Graphics (ToG)*, 30(3), 2011. 32, 63, 63, 68, 68, 69, 70, 84, 87, 130, 146

[UFF06] Raquel Urtasun, David J. Fleet, and Pascal Fua. 3d people tracking with gaussian process dynamical models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 23, 26

[UFHF05] Raquel Urtasun, David J. Fleet, Aaron Hertzmann, and Pascal Fua. Priors for people tracking from small training sets. In *International Conference on Computer Vision (ICCV)*, 2005. 17, 23

[VBK05] Sundar Vedula, Simon Baker, and Takeo Kanade. Image-based spatio-temporal modeling and view interpolation of dynamic events. *ACM Transactions on Graphics (ToG)*, 24(2), 2005. 32

[VHKK12] Anna Vögele, Max Hermann, Björn Krüger, and Reinhard Klein. Interactive steering of mesh animations. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2012. 125

[VL10] Jack Valmadre and Simon Lucey. Deterministic 3d human pose estimation using rigid structure. In *European Conference on Computer Vision (ECCV)*, 2010. 27, 27, 28

[VSJ08] Marek Vondrak, Leonid Sigal, and Odest Chadwicke Jenkins. Physical simulation for probabilistic motion tracking. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 26

[WC09] Xiaolin K. Wei and Jinxiang Chai. Modeling 3d human poses from uncalibrated monocular images. In *International Conference on Computer Vision (ICCV)*, 2009. 27, 27, 27, 28

[WC10] Xiaolin Wei and Jinxiang Chai. Videomocap: modeling physically realistic human motion from monocular video sequences. *ACM Transactions on Graphics (ToG)*, 2010. 26, 26, 28

[WFH08] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(2), 2008. 24

[Wik13] Wikipedia. Horse gait, 2013. `http://en.wikipedia.org/wiki/Horse_gaits`. 128, 128

[WL13] Fang Wang and Yi Li. Beyond physical connections: Tree models in human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 116, 118

[WTR11] Xiaomao Wu, Maxime Tournier, and Lionel Reveret. Natural character posing from a large motion database. *IEEE Computer Graphics and Applications*, 2011. 32

[WV03] Jane Wilhelms and A. Van Gelder. Combining vision and computer graphics for video motion capture. *The Visual Computer*, 2003. 33

[WWL$^+$14] Chunyu Wang, Yizhou Wang, Zhouchen Lin, Alan L. Yuille, and Wen Gao. Robust estimation of 3d human poses from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. xii, 16, 16, 17, 27, 29, 29, 30, 32, 45, 86, 114, 114, 115, 115

[YAH10] Katsu Yamane, Yuka Ariki, and Jessica Hodgins. Animating non-humanoid characters with human motion data. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2010. 125

[YGVG12] Angela Yao, Juergen Gall, and Luc Van Gool. Coupled action recognition and pose estimation from multiple views. *International Journal of Computer Vision (IJCV)*, 2012. 26, 26, 27

[YKC13] Tsz-Ho Yu, Tae-Kyun Kim, and Roberto Cipolla. Unconstrained monocular 3d human pose estimation by action detection and cross-modality regression forest. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 17, 23, 26, 26, 28

[YKW13] Hashim Yasin, Björn Krüger, and Andreas Weber. Model based full body human motion reconstruction from video data. In *6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications (MIRAGE)*, 2013. xv, xv, 73, 73, 73, 73, 74, 74, 75, 84, 84, 94, 108

[YR11] Yi Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. xvii, xvii, 21, 21, 21, 21, 29, 31, 32, 81, 99, 101, 102, 106, 116, 118

[YVN$^+$14] Innfarn Yoo, Juraj Vanek, Maria Nizovtseva, Nicoletta Adamo-Villani, and Bedrich Benes. Sketching human character animations by composing sequences from large motion database. *The Visual Computer*, 30(2), 2014. 30, 31

[ZLCK05] Jianhui Zhao, Ling Li, and Kwoh Chee Keong. 3d posture reconstruction and human animation from 2d feature points. *Computer Graphics Forum*, 2005. 27, 29

[ZLLS14] Liuyang Zhou, Zhiguang Liu, Howard Leung, and Hubert P. H. Shum. Posture reconstruction using kinect with a probabilistic model. In *20th ACM Symposium on Virtual Reality Software and Technology (VRST)*, 2014. 31, 31