Universitat Politècnica de Catalunya

Articulated Models for Human Motion Analysis

Ph.D. Thesis

Author:
Adolfo López Méndez

Thesis Advisor:
Josep R. Casas

October 2012

Oficina de Doctorat
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Acta de qualificació de tesi doctoral

Curs acadèmic:

Nom i cognoms

DNI / NIE / Passaport

Programa de doctorat

Unitat estructural responsable del programa

Resolució del Tribunal

Reunit el Tribunal designat a l'efecte, el doctorand / la doctoranda exposa el tema de la seva tesi doctoral titulada

_____

_____.

Acabada la lectura i després de donar resposta a les qüestions formulades pels membres titulars del tribunal, aquest atorga la qualificació:

☐ APTA/E ☐ NO APTA/E

(Nom, cognoms i signatura) President/a		(Nom, cognoms i signatura) Secretari/ària
(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal

_____, _____ d'/de _____ de _____

El resultat de l'escrutini dels vots emesos pels membres titulars del tribunal, efectuat per l'Escola de Doctorat, a instància de la Comissió de Doctorat de la UPC, atorga la MENCIÓ CUM LAUDE:

☐ SI ☐ NO

(Nom, cognoms i signatura) Presidenta de la Comissió de Doctorat	(Nom, cognoms i signatura) Secretària de la Comissió de Doctorat

Barcelona, _____ d'/de _____ de _____

A la meva família
i als amics que he fet durant aquesta etapa.

"All our knowledge has its origins in our perceptions."

Leonardo da Vinci

Summary

Human motion analysis is as a broad area of computer vision that has strongly attracted the interest of researchers in the last decades. Motion analysis covers topics such as human motion tracking and estimation, action and behavior recognition or segmentation of human motion. All these fields are challenging due to different reasons, but mostly because of viewing perspectives, clutter and the imprecise semantics of actions and human motion.

The computer vision community has addressed human motion analysis from several perspectives. Earlier approaches often relied on articulated human body models represented in the three-dimensional world. However, due to the traditionally high difficulty and cost of estimating such an articulated structure from video, research has focus on the development of human motion analysis approaches relying on low-level features. Although obtaining impressive results in several tasks, low-level features are typically conditioned by appearance and viewpoint, thus making difficult their application on different scenarios. Nonetheless, the increase in computational power, the massive availability of data and the irruption of consumer-depth cameras is changing the scenario, and with that change human motion analysis through articulated models can be reconsidered.

Analyzing and understanding of human motion through 3-dimensional information is still a crucial issue in order to obtain richer models of dynamics and behavior. In that sense, articulated models of the human body offer a compact and view-invariant representation of motion that can be used to leverage motion analysis. In this dissertation, we present several approaches for motion analysis. In particular, we address the problem of pose inference, action recognition and temporal clustering of human motion. Articulated models are the *leitmotiv* in all the presented approaches. Firstly, we address pose inference by formulating a layered analysis-by-synthesis framework where models are used to generate hypothesis that are matched against video. Based on the same articulated representation upon which models are built, we propose an action recognition framework. Actions are seen as time-series observed through the articulated model and generated by underlying dynamical systems that we hypothesize that are generating the time-series. Such an hypothesis is used in order to develop recognition methods based on time-delay embeddings, which are analysis tools that do not make assumptions on the form of the form of the underlying dynamical system. Finally, we propose a method to cluster human motion sequences into distinct behaviors, without a priori knowledge of the number of actions in the sequence. Our approach relies on the articulated model representation in order to learn a distance metric from pose data. This metric aims at capturing semantics from labeled data in order to cluster unseen motion sequences into meaningful behaviors. The proposed approaches are evaluated using publicly available datasets in order to objectively measure our contributions.

Resum

L'anàlisi del moviment humà és una àrea de visió per computador que, en les últimes dècades, ha atret l'interès de la comunitat científica. L'anàlisi de moviment inclou temes com el seguiment del cos humà, el reconeixement d'accions i patrons de comportament, o la segmentació del moviment humà. Tots aquests camps suposen un repte a causa de diferents raons, però especialment a la perspectiva de captura de les escenes a analitzar i també a l'absència d'una semàntica precisa associada a les accions i el moviment humà.

La comunitat de visió per computador ha abordat l'anàlisi del moviment humà des de diverses perspectives. Els primers enfocaments es basen en models articulats del cos humà. Aquests models representen el cos com una estructura esqueletal tridimensional. No obstant, a causa de la dificultat i el cost computacional de l'estimació d'aquesta estructura articulada a partir de vídeo, la investigació s'ha anat enfocant, en els últims anys, cap a l'anàlisi de moviment humà basat en característiques de baix nivell. Malgrat obtenir resultats impressionants en diverses tasques, les característiques de baix nivell estan normalment condicionades per l'aparença i punt de vista, cosa que fa difícil la seva aplicació en diferents escenaris. Avui dia, l'augment de la potència de càlcul, la disponibilitat massiva de dades i la irrupció de les càmares de profunditat de baix cost han proporcionat un escenari que permet reconsiderar l'anàlisi de moviment humà a través de models articulats.

L'anàlisi i comprensió del moviment humà a través de la informació tridimensional segueix sent un enfocament crucial per obtenir millors models dinàmics al voltant del moviment del cos humà. Per això, els models articulats del cos humà, que ofereixen una representació compacta i invariant al punt de vista de la captura, són una eina per potenciar l'anàlisi de moviment. En aquesta tesi, es presenten diversos enfocaments per a l'anàlisi de moviment. En particular, s'aborda el problema de l'estimació de pose, el reconeixement d'accions i el clustering temporal del moviment humà. Els models articulats són el leitmotiv en tots els plantejaments presentats. En primer lloc, plantegem l'estimació de pose mitjançant la formulació d'una mètode jeràrquic d'anàlisi per síntesi en què els models s'utilitzen per generar hipòtesis que es contrasten amb vídeo. Fent servir la mateixa representació articulada del cos humà, es proposa una formulació del moviment humà per al reconeixement d'accions. La nostra hipòtesi és que les accions formen un conjunt de sistemes dinàmics subjacents que generen observacions en forma de sèries temporals. Aquestes sèries temporals són observades a través del model articulat. Aquesta hipòtesi s'utilitza amb la finalitat de desenvolupar mètodes de reconeixement basats en time-delay embeddings, una eina d'anàlisi de sèries temporals que no fa suposicions sobre la forma del sistema dinàmic subjacent. Finalment, es proposa un mètode per segmentar seqüències de moviment del cos humà en diferents comportaments o accions, sense necessitar un coneixement a priori del nombre d'accions en la seqüència. El nostre enfocament utilitza els models articulats del cos humà per aprendre una distància mètrica. Aquesta mètrica té com a objectiu capturar la semàntica implícita de les anotacions que es puguin trobar en altres bases de dades que continguin seqüències de moviment. Amb la finalitat de mesurar objectivament les nostres contribucions, els mètodes proposats són avaluats utilitzant bases de dades públiques.

Agraïments

Vull agraïr al meu director de tesi, Josep R. Casas, la seva dedicació durant els anys que ha portat fer aquesta tesi. I, evidentment, no puc oblidar-me dels companys del Grup de Processament de Imatge i Vídeo, pels debats, articles, pastissos, demos i hores compartides. Entre aquests no puc deixar de mencionar al Marcel Alcoverro, ja que el seu treball es veu reflectit en una part important d'aquesta tesi. I would also thank Juergen Gall and the people from BIWI at ETH Zurich for making my visit valuable and enjoyable.

Contents

List of Figures

List of Tables

1

Introduction

Human motion analysis is as a broad area of computer vision that has strongly attracted the interest of researchers in the last decades. Motion analysis covers topics such as human motion tracking and estimation, action and behavior recognition or segmentation of human motion. All these fields are challenging due to different reasons, but mostly because of viewing perspectives, clutter and the imprecise semantics of actions and human motion. In spite of that, the wide range of applications in several areas and disciplines, such as animation, gaming, data indexing and retrieval, surveillance or biomechanics, to name a few, have pushed researchers forward to develop systems facing with these challenges.

The computer vision community has addressed human motion analysis from several perspectives. Earlier approaches tend to rely on models of the human body that contained real-world magnitudes such as 3D positions. Markerless motion capture approaches are possibly one of the most salient examples of approaches that deal with human pose estimation and motion analysis by means of real-world magnitudes, represented by an articulated model. However, estimating 3D magnitudes from images is a highly challenging problem in most cases. This limitation has led to the development of human motion analysis approaches relying on low-level features. Although obtaining impressive results in several tasks, low-level features are typically conditioned by appearance and viewpoint, thus making difficult their application on different scenarios. Hence, it seems

that the understanding of human motion through 3-dimensional information is still a crucial issue in order to obtain richer models of dynamics and behavior.

1.1 A look into the past

Understanding the movement of humans, both in terms of how it is generated and how it is interpreted, has been of interest in several disciplines, such as science or arts. This interest goes back very far in history. Classical antiquity can be regarded as the seed period for the most sounding technical contributions for human motion understanding. In this period, we can find works that expose mathematical models and studies for human poses and motion for different activities [KT08]. These works were usually reflected through arts, especially in the case of sculpture, where artists looked for a highly realistic representation of motion through a single static *frame*.

Renaissance continued these trends. The sketches and works of Leonardo Da Vinci (1452-1519) contained studies about human body and motion. He actually coined the term *kinematic trees*, referring to the kinematic chains that model the underlying structure of the human body, and that allow to represent human poses by means of articulated models. It is worth mentioning that the artistic currents of the Renaissance also focused on the thorough development of the perspective as a tool for representing scenes in a single *frame* or picture. Therefore, artists of that time were actually concerned about the representation of the 3-dimensional world and its relationship to the appropriate representation of human pose and its motion.

Later on, Giovanni Alfonso Borelli (1608-1679) contributed to human motion understanding with studies that applied mechanics to analyze motion for biological purposes. His main contribution was a leap between qualitative observational studies to quantitative measurements of motion. Such a work is considered to be crucial for the birth of biomechanics [KT08].

After Borelli, the most notable contributions to the understanding of human motion are found in the XIX century. These contributions came along with revolutionary technical advancements in capturing devices, and more specifically the chronophotography. Etienne-Jules Marey (1830-1904) is one of the best representatives of the use of chronophotography for the study of motion. He actually invented a system that can be considered the first marker-based motion capture system 1.1. Inspired by the work of Marey, Muybridge developed a system to record fast motion with several cameras. He also invented a system to display the recorded images, hence he was influential in the development of the cinematography. He used these technical contributions to make

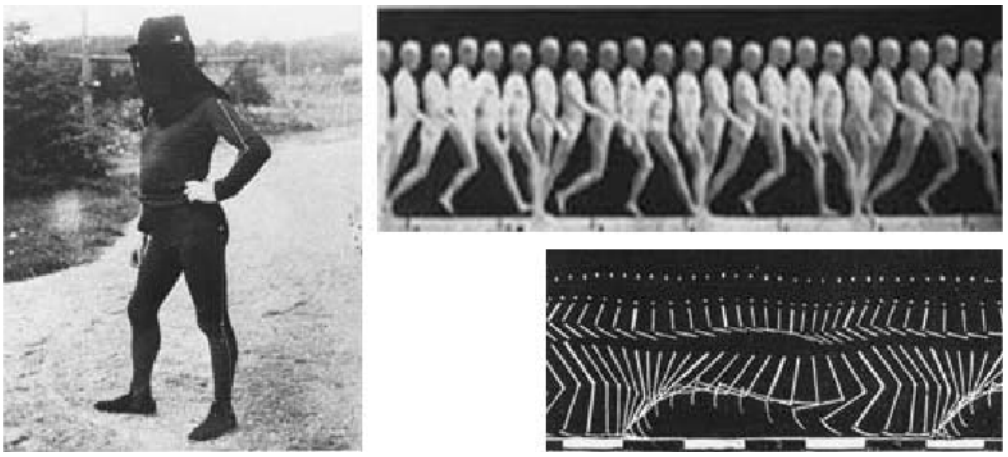


FIGURE 1.1: The system developed by Etienne-Jules Marey in order to capture human motion [Mar94]

locomotion studies of humans. The images recorded during these studies have been used in modern computer vision publications [BM98].

The origins of computer-based human gait analysis are probably found in the work of Gunnar Johansson [Joh73]. He studied gait by recording humans having a set of light displays attached to actor's limbs. The resulting configuration is rather similar to the one employed in modern marker-based motion capture systems.

1.2 Recent Years and Future Perspectives

The availability of cameras and the proliferation of the processing power of modern computers has paved the way to the development of computer vision technologies for the understanding of human motion. Motion understanding, action recognition and pose estimation are central problems in computer vision, computer graphics and image processing. However, in most cases, the historical link between perspective and human body modeling has been broken due to the inherent complexity of the problem of estimating 3D information from images. Consequently, a large number of approaches towards human motion and behavior understanding have focused on the dynamical analysis of low-level and appearance-based features. However, recent technological advances have changed the scenario. Firstly, the constant increase in computational power allows researches to deal with large-scale vision problems. At the same time, this computational power, along with global communication, has produced a massive amount of data that can be used in research in order to produce better models of vision. Finally the recent release of consumer-depth cameras, probably led by the Microsoft's Kinect device [Kin] is a solution to some of the chronic problems of vision in order to estimate the human pose.

All these advances are pushing forward the historical paradigm of motion understanding: the human body is represented by an underlying skeletal model comprising joints linked by bones and this model can be properly represented (estimated) through perspective representation. Hence, there is a promising near future for approaches relying on articulated body models.

1.3 The role of articulated models in the understanding of human motion

In this thesis, we investigate the potential of articulated body models for human motion analysis and understanding. We refer to motion analysis in a wide sense and our work ranges from pose inference to discovery of action categories in a human motion sequence. Articulated models are a *sketch* of the human body and its movement. Compared to low-level features, articulated models can efficiently represent complex poses and motion with a relatively low number of parameters or dimensions. Models are a tool for representing prior knowledge about the body structure. They also define a splitting of feature vectors into global rotations and translations and local joint rotations. This splitting helps in easily representing poses and motion independently of a viewpoint. Skeletal models allow also computing symmetric configurations, which may be of help in order to learn patterns of motion that can be performed with left or right limbs.

All these features are exploited in this thesis in order to provide a human motion understanding framework robust to view-point and appearance changes. The following sections describe our research on several fields of human motion understanding through the use of articulated body models.

1.4 Contribution

This thesis focuses on the use of articulated models for the analysis of human motion from multiple views or eventually with range sensors. Human motion analysis is performed in three different levels with the following contributions:

1. **3D Pose estimation** : Estimating the human pose is crucial for further motion and behavior analysis. In this thesis, we mainly focus on inferring pose from multiple views. To this end, we adopt an approach whose central element is a human body model governed by an articulated skeletal model. Our contributions are:

- A layered particle filtering approach that generalizes existing approaches of the state-of-the-art.
- A detector-driven hierarchical particle filter, that is designed within our layered framework, providing state-of-the-art performance without learning motion models.
- A method to detect 3D body parts from multiple views.
- An approximate partitioning of the observations, which are images in multiple views or range images. Such partitioning helps in improving the observation model and provides a more accurate and faster pose inference.

2. **Action Recognition** : Given labeled sequences of poses, represented by means of an articulated model, our aim is to recognize actions performed in new sequences. To this end, we propose a principled approach towards the characterization of action sequences as time series. Our contributions in this field are:

   - A novel approach towards exploiting time-delay embeddings. Time-delay embeddings are a principled method to reconstruct dynamical systems from data. Our approach consists in taking advantage of the geometry and topology of these systems in order to perform action recognition. For this reason, we propose reconstructing several dynamical systems in the same space in order to be compared.
   - A distance between reconstructed phase spaces, that takes ideas from dynamic programming in order to find a topologically suitable alignment between phase-space trajectories. We use this distance with two classification strategies in order to recognize actions from human motion sequences.
   - An action classification method based on constructing codebooks with delay vectors. The classification approach is inspired in the popular bag-of-words model, but in this thesis words implicitly encode dynamical information.
   - An efficient random forest approach to classify actions in the phase space reconstructed by time-delay embeddings.

3. **Temporal Clustering of Human Motion** : We are given unseen motion sequences containing a set of actions. Assuming that we have a set of examples whose labels do not appear in the unseen actions, but are related to some extent, we would like to cluster the unseen sequences into different behaviors. This problem can be formulated as a weakly supervised temporal clustering of motion sequence. Our contributions are the following:

   - An approach to learn *what makes some actions different from others* by means of metric learning. The novelty of this approach is that the ultimate goal is

to apply the learned metric to unseen data, containing possibly unseen action categories. Hence, we expect the metric to gather some semantics in order to quantize unseen motion sequences into meaningful motion primitives. To this end, we propose a set of constraints specifically suited for articulated models.

- A framework that combines metric learning and hierarchical Dirichlet processes in order to cluster motion sequences into actions or behaviors. Our approach has the advantage that can be used across datasets without providing the number of clusters or actions.

An in-depth list of the contributions of this thesis, including related references of journal papers, international conference publications, contributions to projects and submitted publications is detailed in the conclusions chapter.

1.5 Thesis Organization

The thesis is structured as follows:

- Chapter 2 provides an overview of the state-of-the-art in pose estimation, human motion and behavior understanding.
- Chapter 3 is devoted to pose inference. In this chapter, we present our layered approach to the model-based pose inference problem.
- Chapter 4 presents the human action recognition problem as the analysis of articulated motion as a set of time series. We describe principled approaches to the analysis of such time series in order to classify human actions.
- Chapter 5 deals with the problem of identifying distinct human behaviors in motion sequences. Differently from Chapter 4, models for specific actions, learned from labeled data, cannot be used, since one has to deal with unseen actions or strong stylistic changes. Furthermore, the approach presented in this chapter is shown to work reliably across datasets.
- Chapter 6 draws the conclusions of this dissertation.

2

State of the Art

In the recent years, human motion understanding from video streams has become a foremost issue in computer vision and image processing. Human motion understanding is itself a broad area comprising pose inference, action recognition, motion segmentation and other related fields. These fields have been traditionally tackled with tools and methods that work independently on each problem. For that matter, in the following we present state-of-the-art approaches that address the problems covered in this dissertation: pose inference, action recognition and segmentation of human motion motion into distinct and semantically meaningful behaviors.

2.1 Pose Inference

In this dissertation we focus on 3D pose inference from video, that is, the automated estimation of the configuration of a three-dimensional articulated model that represents the underlying structure the human body. Even addressed by analyzing multiple views, the problem of pose inference is often ill-posed due to spatial ambiguities, mainly caused by self-occlusions. In addition, in realistic conditions, the variability in individuals' appearance and background clutter make the estimation much more difficult.

Approaches addressing pose inference can be split into 2 main categories: example-based and model-based.

Example-based pose inference methods aim at finding a direct mapping between the feature space and the 3D pose space. One of the earliest attempts to example-based pose inference was the nonlinear regression against shape descriptors automatically obtained from silhouettes in monocular images [AT06]. Similarly, Elgammal and Lee [EL04] proposed a nonlinear mapping from silhouettes in multiple views to 3D poses. Using Locally Linear Embedding [RS00] and Isomap [TSL00], they embed the silhouette information in a low-dimensional space while keeping some degree of the geometrical structure of the manifold. By connecting the points in the manifold, they reveal motion trajectories that are view-dependent. From these view dependent manifolds, they learn a mapping in order to infer the 3D pose parameters. Howe et. al. [How04] proposed a direct silhouette lookup based Chamfer distance to select candidate poses. Because pose to image is a many-to-one mapping, some temporal constraints are added in order to improve the pose inference. Gaussian Processes Latent Variable Models [Law04] and Gaussian Process Latent Dynamical Models [WFH05] are another kind of mappings that involve a latent low-dimensional space that is mapped onto the input space. These approaches have been used for human body tracking of a priori modeled actions [UFF06]. More recently, a solution to the problem of having to learn a huge number of poses has been proposed in [UD08]. This technique is based on a local mixture of Gaussian processes, where locality is defined based on both appearance and pose. The mixture components are defined online in very small neighborhoods, so learning and inference is extremely efficient.

On the contrary, model-based approaches make use of the kinematic constraints imposed by an articulated model. These constraints range from simple limitations such as *the arm is connected to the shoulder* to more precise constraints such as *the elbow angle is in the interval* $[-\pi, 0]$. Similarly to other graphical models used in different computer vision problems, articulated body models act as a prior of the structure and motion of the human body and hence have the role to make pose inference from video more tractable. If the articulated model is *fleshed* with some shape model (volumetric primitives or a polygonal mesh representing the human body), then body models act also as priors for the appearance of the human body. We further categorize model-based approaches depending on whether the model is used implicitly to represent some kinematic constraints, or it used explicitly, thus also representing the human body shape.

When implicitly used, the human body model is used a graphical model in order to formulate the pose estimation problem. Some authors perform inference on this graphical

model by means of Belief Propagation [Mac03]. Sigal et al. [SBR+04] proposed a loose-limbed model of people in order to infer poses with a belief propagation method. More recently, Bernier et. al [BCMCB09] have proposed a nonparametric belief propagation method for upper-body tracking with stereo images. In [BKSS10], Bergthold et al. propose a graphical model framework with application to 3D pose estimation. Their modeling framework implicitly represents a skeletal model comprising a set of joints, although they formalize kinematic constraints through a fully connected graph.

Approaches using a human body model explicitly fall in the category of analysis-by-synthesis, since the modeling framework serves to generate pose hypothesis that are represented by human shaped models. Withing analysis-by-synthesis approaches, stochastic sampling, and more concretely particle filters (PF) [AMG+02] [IB98] have become predominant methods due to their ability to deal with non-linear and non-Gaussian processes. Nevertheless, the large number of degrees of freedom that can be found in an articulated body model makes the body tracking problem a high-dimensional state-space problem. Particle Filters are a good approach for tracking in low dimensional spaces, but they become inefficient in high-dimensional problems, because they require a number of particles that grows exponentially with the number of dimensions. For that matter, a number of variants addressing the curse of dimensionality have been proposed [DBR00, DR05, HWG07, GRBS10, BB09, BJB12]. For a more in-depth analysis of these approaches, we refer the reader to Chapter 3.

2.2 Action Recognition

In computer vision, a human action can be described as a simple motion pattern usually lasting a short period of time and typically executed by a single person[TCSU08].

Earlier approaches to human action recognition relied on three-dimensional articulated models and kinematic features [CB95, PR00, BCMS01]. These approaches were gradually abandoned due to the difficulty of estimating 3D poses from video. Human action recognition methods relying on low-level features emerged as a promising alternative. These methods provide accurate results using descriptors than can be easily computed from video. In this dissertation we use the term model-free to denote these approaches.

One of the earliest attempts of model-free action recognition are the motion templates proposed by Bobick and Davies [BD01]. By gathering a temporal sequence of motion areas computed with human silhouettes, they were able to recognize several actions. The original proposal is view dependent as it relies on silhouettes extracted from a single

view. The idea of extending these temporal templates to volumetric reconstructions [WBR07, CFCP06] proved to be feasible.

Following the success of interest point detectors for image registration and object detection, Laptev et al. propose the Space-Time Interest Points (STIP) [LL03]. Similarly, Dollar et al. [DRCB05] introduce sparse spatio-temporal interest points for action recognition. Both approaches succeed in obtaining informative points in order to compute features for action recognition from low-resolution videos. Niebles et al. [NWFf06] use STIPs to build a bag-of-words model of actions in video. More recent approaches [LMS+, KG10] also rely on STIPs in order to compute video descriptors such as 3D-SIFT [SAS07], HOG3D [KMS08], extended SURF [WTvG08] or local trinary patterns [YW09]. Alternatively, Jhuang et al. [JSWP07] propose spatio-temporal filtering of videos. This approach has been described as a model of the visual cortex. More recently, extracting dense trajectories from video has been shown to outperform sparse spatio-temporal interest points [WKSL11]. For a deeper analysis of model-free methods, we refer the reader to [TCSU08, Pop10].

In the recent years, the computer vision community has made much progress in 3D pose inference methods. The reasons of such a progress are the increase of computational power, the massive availability of video and motion capture data and the evolution of capturing devices, and more specifically consumer depth cameras [Kin, Xti]. All these elements configure a suitable scenario to reconsider the earlier model-based approaches for human action recognition.

Model-based approaches are able to tackle the problem of view-dependency whenever the model can be represented in the 3-dimensional world in a view-invariant manner [PC06]. This is the case of virtual skeleton representations, where motion sequences can be easily represented independently of the overall rotation and translation parameters.

Within model-based approaches, this dissertation is related to the analysis of time series and the underlying dynamical systems that describe the motion of joints in a human model. Pavlovic and Rehg [PR00] infer the kinematics of walking and running to model dynamical systems that can be used to recognize both actions. With similar kinematic features, Bisacco et al. [BCMS01] classify human gaits by defining a distance in the space of dynamical systems. Campbell and Bobick [CB95] represent human motion using joint angles to construct curves in a phase space. They are able to segment fundamental steps in ballet dance. Lv et al. [LNL05], infer human actions by representing them as a set of weighted channels consisting in the temporal evolution of 3D joint coordinates. Raptis et al. [RBS07] compare time series by considering a time warping model that accounts for the intrinsic characteristics of an underlying dynamical system. More recently, Raptis et al. [RWS10], have proposed a method to model human actions as a linear time-invariant

dynamical model of relatively small order that generates multivariate time series in the form of joint angle dynamics along time.

Some authors focus on dimensionality reduction methods to deal with the action recognition problem by embedding input feature spaces into low-dimensional spaces. Dimensionality reduction methods have been widely applied also in the field of model-free action recognition [CWSS07, WS07, BR07], due to the high dimensionality of many image features. However, when one deals with actions and time series, keeping the geometric structure of input spaces is often not sufficient, and there is a need for constraining embedding techniques to cope with the underlying dynamics. Lewandowski et al. [LMdRMN10], propose Temporal Laplacian Eigenmaps to embed time series of motion capture data and video data, obtaining good results on both human motion reconstruction and action recognition.

A recent approach by Yao et al. [YGvG12] presented a coupled markerless motion capture [GYvG10] and action recognition [YGFvG11] system. This work shows the feasibility of model-based action recognition, and its superior performance in front of low-level features. The work by Guerra-Filho and Aloimonos [GFA12] is related to action recognition, but they give one step further by analyzing the underlying grammars of human motion that define actions. Their work is based on the analysis of mocap data, and thus, of articulated models.

2.3 Segmentation of Motion into Behaviors

Although begin a challenging problem, in posing action recognition a direct relation between the perception of motion and semantics is established: action labels are given for each action sequence. Although intra-class variations can make this link rather elusive, a more challenging problem to be solved is that of finding semantically meaningful action categories from videos or motion sequences. This problem implies that precise labeling of actions is not given. For that reason it is a very difficult problem, that is even challenging for humans: given a motion sequence or a video, different people might perceive different action categories.

In order to efficiently annotate actions in large collections of video or mocap data, some researchers have focused on unsupervised segmentation and clustering of human actions. Niebles et al. [NWFf06] propose a bag-of-words model based on STIPs (see previous section) in order to model human motion in video. Using probabilistic Latent Semantic Analysis (pLSA) they find different action categories in videos. As far as mocap data is concerned, Barbic et al. [BSP+04] propose a change detection algorithm

that provides accurate segmentation results, but that is unable to cluster the temporal segments into the different behaviors. Ozay et al. [OSC08] overcome this problem by modeling the first three principal components of the data as an autoregressive model. The coefficients of the model are then clustered with k-means. Similarly, the Aligned Cluster Analysis (ACA) proposed by Zhou et al. [ZDH08] extends the k-means concept to cluster time series. They show that ACA can accurately find different behaviors in sequences of video and mocap data. However, both [OSC08] [ZDH08] require to manually set the number of clusters (actions) k. In [RWS10], this limitation is tackled by using a spike-train driven dynamical model that can detect motion transitions and clusters them into different behaviors, without having to manually set the number of clusters k. Recent approaches such as [KOSS11][KBvGF10] have proposed variants and extensions of hierarchical Dirichlet processes (HDP) [TJBB06] in order to find activities using optical flow features mainly. In [FSJW08], HDPs are used as a prior for HMM parameters in order to cluster time series data into distinct behaviors. This latter approach is applied to synthetic data, stock indices and dancing honeybee data.

3

Layered Pose Inference

3.1 Introduction

Pose inference from video data, also known as markerless motion capture or, in other words, being able to automatically estimate the human pose and its motion, is a problem that has attracted the interest of the computer vision community in the last decades. Among its wide range of applications, we emphasize its potential for action recognition [HTTM11], a problem of motion understanding with which we deal in Chapter 4.

In this chapter, we present a layered framework for pose estimation from multiple views or range data. The proposed layered framework is initially derived as a solution for well-known problems of state-space dimensionality when using particle filters. We show that this framework is in fact a generalization of existing particle filtering methods that take into account the curse of dimensionality. However, we allow an increased flexibility in the definition of layers that allows, for instance, multiple passes on state-space variables. In addition, in the absence of actual likelihood functions for pose estimation, we relate particle filtering to optimization by means of cost functions. This relation, already established in other works, leads us to combine particle filtering (understood as a global stochastic optimization method) with local optimization in order to refine the posterior mean estimate.

We start by introducing particle filtering in order to extend it to a layered filtering scheme. Once layered particle filtering has been presented, we show how to instantiate such a framework in order to implement existing particle filtering approaches towards motion capture. We then advocate the use of hierarchically organized layers (according to the implicit hierarchical structure of the articulated model). This choice is motivated using graphical models, but is also shown to outperform other strategies in several experiments. More importantly, hierarchical layers are convenient in order to define a novel propagation approach that employs body part detectors. Not only propagation can be efficiently improved in a hierarchically layered particle filter, but also the observation model. We propose a simple yet effective method to approximately partition the observations in order increase the efficiency and accuracy of pose inference. All these contributions are thoroughly evaluated in publicly available datasets and compared with the state-of-the-art.

3.2 Related Work

Pose inference has been addressed from several perspectives. Approaches towards pose inference can be split into two big categories, depending on the use of a human body model. If no model is used, then we refer to example based inference methods. Such approaches aim at learning a mapping between the image space and the pose space [AT06, LE10, UD08]. On the other hand, model-based approaches make an implicit or explicit use of a human body model, a kinematic structure comprising joints linked by edges [DBR00, GRS08, GRBS10, BB09]. The work presented in this thesis belongs to the second group, and more specifically, it makes an explicit use of a human body modeling framework for analysis-by-synthesis inference of pose. This sort of approaches are usually employed in multi-camera scenarios or with range data.

Despite recent advances, pose inference from multiple views is still a challenging problem in realistic scenarios, where a few views are usually available. Some authors employ a model-based approach in combination with variants of particle filter algorithms in order to efficiently estimate the pose [GRBS10, BB09]. Although achieving an impressive performance on some standard datasets, they usually rely on weak prior models that fail to predict complex motion. For that matter, learning motion models has been proposed as a solution to enrich such prior models . Some approaches rely on prior models built on low-dimensional pose spaces and manifolds. Such spaces can be obtained by linear dimensionality reduction techniques [SBF00, SBS02, UFHF05] or non-linear dimensionality reduction techniques [JKMvG09, LTSY10, GYvG10]. Other prior models are based on probabilistic latent variable models [WFH08, TSFH10]. The problem with

these approaches is the lack of scalability when a large number of motion patterns need to be tracked or the poor generalization in front of unseen motion categories. Alternatively, one can rely on input video cues in order to improve prior models. This latter approach can be addressed by detecting body parts on images [SNH10]. Body part detectors are also the basis for part-based graphical models. In [BKSS10], Bergthold et al. propose a graphical model framework with application to 3D pose estimation. Unary potentials can be seen as the response of body part detectors based on several features. However, even some existing approaches successfully deal with 3D pose estimation with body part detectors, employing image cues and body part detectors is frequently overlooked in multiple view pose estimation due to the difficulty of robustly fusing detections in several views.

The layered framework presented in this dissertation has some similarities to the partitioned approach proposed by Bandouch et al. [BEB08, BB09, BJB12]. In agreement with their work, we experimentally found that hierarchical layers are more convenient that exploring the whole state-space with meta-heuristics such as simulated annealing [DBR00]. Differently to the approaches by Bandouch et al., we propose not limit the layers to partitions, but to include any subset that holds a hierarchical ordering with respect to the structure of the articulated model. We also propose different propagation and observation models for hierarchical layers, and we advocate the use of local optimization for pose refinement.

3.3 Approach Overview

The pose inference problem can be understood as estimating the positions of body joints in a skeletal model representing the human body. Provided that we implicitly aim at estimating the configuration of a set of joints with a particular topology, we adopt an analysis-by-synthesis approach whose central element is a human body model comprising an articulated structure that consists in the target joints linked by segments that represent human bones. These bones effectively represent the topology of the target joints and, as we will detail later on this chapter, establish an implicit hierarchical relation between them.

This schematic representation of the human body pose allows using the twists and exponential map formulation [BM98] to parameterize human poses as a global translation and rotation and a set of joint rotations. So, in the end, every joint position is determined by these pose variables and the bone lengths. In general, bone lengths are unknown and fall in the category of anthropometric variables. These variables are out

a b c

FIGURE 3.1: Body modeling framework (a) Skeletal Model (b) Mesh Model (c) Wired Mesh model and underlying skeletal model

of the scope of this dissertation, but their estimation has been studied more in depth by Marcel Alcoverro in [ACP10].

Provided that we aim at estimating the configuration of such a model from video, we need some sort of modeling framework allowing us to measure the discrepancy between what we see in images and hypothesis produced by our skeletal model (See Figs 3.1a). Such a requirement is fulfilled by a shape model usually implemented by means of simple yet efficient sets of volumetric primitives, or more sophisticated polygonal meshes representing the shape of the body (See Figs 3.1b, c).

Two scenarios are of interest in our work: multi-camera settings, where several cameras are placed in a wide baseline setup, and single view settings, with a range sensor [Kin, Xti, Pri], though in this chapter (and in the dissertation in general) we make emphasis on the multi-camera setup.

3.4 Pose Estimation with Layered Particle Filters

Pose estimation can be formulated as a tracking problem that involves estimating the state of pose variables through time. State-space models with non-linear and non-Gaussian transitions are a widely used approach. However, these models lead to analytically intractable statistics that require sub-optimal estimation algorithms. To this end, Monte Carlo methods approximate probability density functions by means of a set of weighted samples. Particle Filters (PFs) [AMG+02] are a particular Monte Carlo method especially tailored to sequences of noisy measurements.

However, regular PF methods usually suffer from the curse of dimensionality: the state space of human model variables is of high dimensionality, thus the number of required particles usually grows exponentially. This problem has led to a generic set of methods that we term *Layered Particle Filters*, since these methods behave as PFs with a refinement procedure based on layers. In the following, we introduce the basic notions for

Particle Filtering to afterwards present the concept of *Layered Particle Filters*. Then, we describe two well-known variants of PFs, the Annealed Particle Filter [DBR00] and several Hierarchical Particle Filters [MH03, HWG07, DD09, BB09], and we show that both are covered by the *Layered Particle Filter* formalism.

3.4.1 Particle Filters

The tracking problem (and thus the pose estimation problem) can be defined by means of a state-space model. Consider a sequence of states over time $\mathbf{x}_t$ constituting the dynamic process. From this sequence, one can recursively find the new state as follows:

$$\mathbf{x}_t = f_t(\mathbf{x}_{t-1}, \mathbf{v}_{t-1}) \tag{3.1}$$

where $\mathbf{v}_{t-1}$ is the process noise and $f_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \rightarrow \mathbb{R}^{n_x}$ is a possibly non-linear state transition function; n_x, n_v are the dimensions of the state vector and the noise of the dynamic process respectively. As a common Bayesian problem, $\mathbf{x}_t$ is a hidden variable producing observations or measurements:

$$\mathbf{z}_t = h_t(\mathbf{x}_t, \mathbf{n}_t) \tag{3.2}$$

where $\mathbf{n}_t$ is the observation noise and $h_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_n} \rightarrow \mathbb{R}^{z_x}$ is a possibly non-linear state transition function; n_z, n_n are the dimensions of the observation vector and the noise of the observation process respectively.

These two equations constitute the Gauss-Markov state-space model that has been adopted to a great extent in tracking problems. The objective of such model is to recursively estimate the degree of belief that a state $\mathbf{x}_t$ is being produced given a collection of observations $\mathbf{z}_{1:t}$ up to time t, i.e, to infer the posterior probability density function (pdf) $p(\mathbf{x}_t|\mathbf{z}_{1:t})$. Note that an initialization $p(\mathbf{x}_0|\mathbf{z}_0) \equiv p(\mathbf{x}_0)$ must be available to recursively estimate the posterior. Then $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ may be obtained by prediction and update. Given the previous posterior $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$, prediction is performed by means of the Chapman-Kolmogorov equation:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1} \tag{3.3}$$

where $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is the state transition prior of a Markov process, which is a commonly adopted assumption for tracking. This probabilistic model is built according to the knowledge of the dynamical process given in equation 3.1.

The update step is the Bayesian computation of the posterior:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \tag{3.4}$$

where the normalizing constant is obtained by the total probability theorem:

$$p(\mathbf{z}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t \tag{3.5}$$

Prediction and update are the formulation to find the optimal recursive Bayes solution for the tracking problem. However, the statistics of a non-linear non-Gaussian process are analytically intractable and therefore we cannot compute the integrals involved in the optimal solution.

A way to deal with complex distributions is by drawing Monte Carlo samples of them [LC98]. Hence, a criterion describing the "goodness" of a sample set is needed:

Definition *A random variable x drawn from a distribution q is said to be* ***properly weighted*** *by a weighting function $w(x)$ with respect to the distribution p if for any integrable function h:*

$$E_q\{h(x)w(x)\} = E_p\{h(x)\}$$

A set of random draws and weights (x^i, w^i), $i = 1...N_s$ is ***properly weighted*** *with respect to p if*

$$\lim_{N_s\to\infty} \frac{\sum_{i=1}^{N_s} h(x^i)w^i}{\sum_{i=1}^{N_s} w^i} = E_p\{h(x)\}$$

for any integrable function h.

According to this definition, a pdf is approximated by discrete distribution of samples with probability proportional to the weights. Following this spirit, Particle Filters (PF) [AMG+02] are designed as recursive Bayesian estimators to approximate the posterior density $p(\mathbf{x}_t|\mathbf{z}_t)$ by means of a set of N_s weighted samples or particles. Given a Bayesian recursive estimation problem with Markov state transitions:

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \tag{3.6}$$

we want to draw a set of properly weighted samples from the posterior. This set can be formulated as follows:

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) \approx \sum_{i}^{N_s} w_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i) \tag{3.7}$$

where w_t^i is the weight associated to the i-th particle. This discrete approximation of the posterior requires the evaluation of weights. This is done by means of the importance sampling principle [DGA00], with a pdf $q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ from which we generate samples that can be evaluated with the posterior (up to proportionality). Applying the importance sampling principle in Eq. 3.6:

$$\begin{aligned} w_t^i &\propto \frac{p(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t})}{q(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t})} \\ w_t^i &\propto \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})q(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t})} p(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1}) \end{aligned} \tag{3.8}$$

and choosing this importance distribution in a way that factors appropriately, we have:

$$\begin{aligned} w_t^i &\propto \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)p(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i,\mathbf{z}_t)q(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})} \\ w_t^i &\propto w_{t-1}^i \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i,\mathbf{z}_t)p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \end{aligned} \tag{3.9}$$

Moreover, if we apply the Markov assumption, the expression is simplified regarding the fact that observations and current state only depend on the previous time instant. Therefore, the PF is a sequential propagation of the importance weights.

A major problem affects the PF. After several iterations the majority of the particles have negligible weights and, as a consequence, the estimation efficiency decays. An effective measure for the particle degeneracy is the survival rate [LC98] given by:

$$\alpha = \frac{1}{N_s \sum_{i=1}^{N_s} (w_t^i)^2} \tag{3.10}$$

In order to avoid this effect, there are two main strategies that can be combined. The first is the choice of the importance distribution. This is crucial since the samples drawn

from $q()$ must be properly weighted with respect to the posterior. It has been shown in [DGA00] that $q(\mathbf{x}_t|\mathbf{x}_{t-1}^i, \mathbf{z}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$ is optimal in terms of variance of the true weights, i.e., the weights obtained by sampling directly the posterior. However, this optimal importance distribution is not always a possible choice.

The second technique consists in resampling the particle set. After likelihood evaluation a new particle set must be drawn from the posterior estimation, hence particles with higher weights are reproduced with higher probability. Once the new set has been drawn all the weights are set to $\frac{1}{N_s}$, leading to a uniformly weighted sample set concentrated around the higher probability zones of the estimated posterior. Resampling is usually applied when the survival rate of the sample set is below a threshold.

The PF estimate should be computed before resampling, because resampling introduces additional random variation. As in Bayesian estimation, there are several options to produce an estimation by means of a significant point of the state-space. Posterior mean, maximum a posteriori or median are valid strategies. Since it is optimal in terms of mean squared error and provides a reduction of the noise introduced when sampling the distributions, the most common option is to use the Monte Carlo approximation of the posterior mean:

$$\hat{\mathbf{x}}_\mathbf{t} = \sum_{i=1}^{N_s} w_t^i \mathbf{x}_t^i \tag{3.11}$$

The Sampling Importance Resampling (SIR) Particle Filter proposed by Gordon et. al [GSS93] is a method commonly used in computer vision problems. It is characterized by applying resampling at every iteration and by defining the importance distribution as the prior or prediction density $p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)$. By substituting this importance density in 3.9:

$$\begin{aligned} w_t^i &\propto w_{t-1}^i \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)p(\mathbf{z}_t|\mathbf{z}_{1:t-1}^i)} \\ w_t^i &\propto p(\mathbf{z}_{1:t}|\mathbf{x}_t^i) \end{aligned} \tag{3.12}$$

Hence, the computation of weights only depends on the likelihood. Consequently, the design of the particle filter is basically a problem of finding an appropriate likelihood function. Since a real likelihood will not be available, we will define likelihood functions as a monotonic decreasing mapping of a cost function. This mapping will be, to some extent, a connection between the filtering and the optimization worlds [GRBS10].

A particular advantage of the SIR PF is that we can easily sample $\mathbf{x}_t^i \sim p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)$ by first generating a noise sample $\mathbf{v}_t^i$ (according to the noise distribution of our system) and then setting $\mathbf{x}_t^i = \mathbf{f}_t(\mathbf{x}_{t-1}^i, \mathbf{v}_t^i)$. The major drawback is that we generate samples without any insight about the observations, thus making the algorithm sensitive to outliers and possibly inefficient. The SIR PF is summarized in 1.

Algorithm 1: SIR Particle Filter

1 Resample particles with replacement $\{\mathbf{x}_{t-1}^i, w_{t-1}^i\} \rightarrow \{\mathbf{x}_t^i, \frac{1}{N_s}\}$
2 Propagate particles by applying a Gaussian diffusion $\mathbf{x}_t^i = \mathbf{x}_t^i + \mathcal{N}(0, \Sigma)$
3 Evaluate particles with the likelihood function $w_t^i \propto p(\mathbf{z}_t|\mathbf{x}_t^i)$
4 Normalize weights so that $\sum_{i=1}^{N_s} w_t^i = 1$
5 Compute the posterior mean estimate $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_s} w_t^i \mathbf{x}_t^i$

3.4.2 Layered Particle Filters

One of the main problems of PFs is the curse of dimensionality: the number of particles needed to efficiently estimate the posterior grows exponentially with the dimensions of the state vector. In pose estimation and tracking, this state space vector is usually of high dimension, meaning that PFs introduced so far are inefficient.

A commonly adopted solution is to divide the filtering in layers. A layered filtering implies performing several PF runs (resampling, propagation and likelihood evaluation) in the same time instant t. In this thesis, we adopt the term *Layered Particle Filtering* to denote those particle filters with a layered structure. As we will show in the following sections, this terminology covers several well-known algorithms in the state-of-the-art and generalizes them by means of a unique conceptual formulation of the algorithm. Moreover, such a formulation has a direct translation on the implementation of such a particle filter.

Without loss of generality, the following formulation of a layered particle filter will be introduced as an extension of the SIR PF: in each layer, the proposal distribution will produce importance weights directly proportional to the likelihood function and resampling will be performed after every layer. These assumptions yield an easier formulation and a simpler way of sampling the posterior distribution.

Before formally describing the *Layered Particle Filters*, let us introduce some notation that will be employed throughout the chapter. Let us denote $\mathbf{x} = \{x_1, x_2, \cdots, x_{nx}\}$ the state-space vector (the same notation employed during the introduction of particle filters) and x_n the one-dimensional variables comprising such a vector. Let us define a

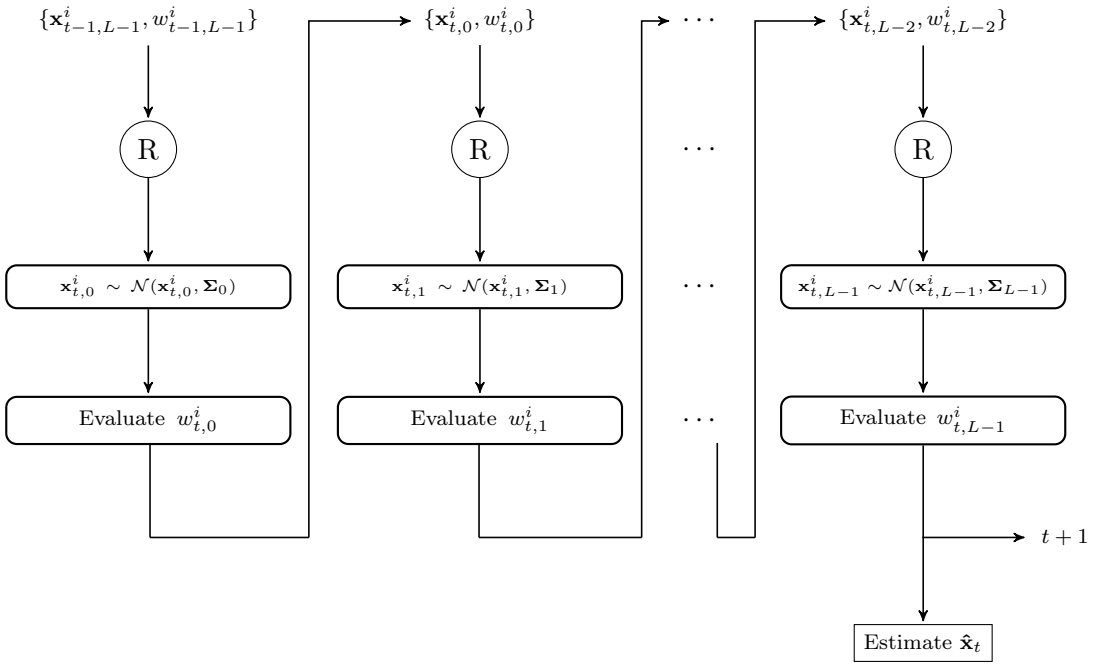


FIGURE 3.2: Layered Particle Filter scheme with Gaussian diffusion in the propagation step.

set of L layers, where the tuple $\{\mathbf{x}^i_{t,l}, w^i_{t,l}\}$ denotes particles and weights in the l-th layer. $\mathbf{x}^i_{t,l}$ denotes the filtering state of the particles in the layer l, thus actually containing all the one-dimensional variables x_n. This filtering state might imply that only a subset of the variables have been filtered up to layer l. Consequently, let $\chi_l = \{x_{\chi_1}, \cdots, x_{\chi_{n_\chi}}\}$ denote a subset of $\mathbf{x}$. Such a subset is associated to the l-th layer, meaning that one-dimensional variables comprising it are incorporated to the filtering process in this layer. Note that these subsets can include all the one-dimensional variables ($\chi_l = \mathbf{x}$), or even the empty set ($\chi_l = \emptyset$) - implying that layer l might not incorporate new variables to the filtering-. Anyhow, subsets must hold:

$$\mathbf{x} = \bigcup_l \chi_l \tag{3.13}$$

In the *Layered Particle Filter*, resampling, propagation and likelihood evaluation are conditioned to some extent to the characteristics of each layer l (see Fig. 3.2).

Likelihood functions in each layer can be conditioned to the subset χ_l in such a way that one might efficiently explore the state space. Formally, the importance weights in each layer l can be formulated as follows:

$$w^i_{t,l} \propto \frac{p(\mathbf{z}_{1:t}|\mathbf{x}^i_{t,l})p(\mathbf{x}^i_{t,l}|\mathbf{x}^i_{t,l-1})}{q(\mathbf{x}^i_{t,l}|\mathbf{x}^i_{t,l-1}, \mathbf{z}_{1:t})} \qquad for\ l > 0 \tag{3.14}$$

$$w^i_{t,0} \propto \frac{p(\mathbf{z}_{1:t}|\mathbf{x}^i_{t,0})p(\mathbf{x}^i_{t,0}|\mathbf{x}^i_{t-1,L-1})}{q(\mathbf{x}^i_{t,0}|\mathbf{x}^i_{t-1,L-1}, \mathbf{z}_{1:t})} \qquad for\ l = 0 \tag{3.15}$$

where the numerator contains the product of the likelihood and the prior, and the denominator contains the proposal distribution. Note that we have assumed Markovian state transitions, resampling in every layer and that the proposal distribution $q()$ factorizes such that we can perform sequential importance sampling within layers.

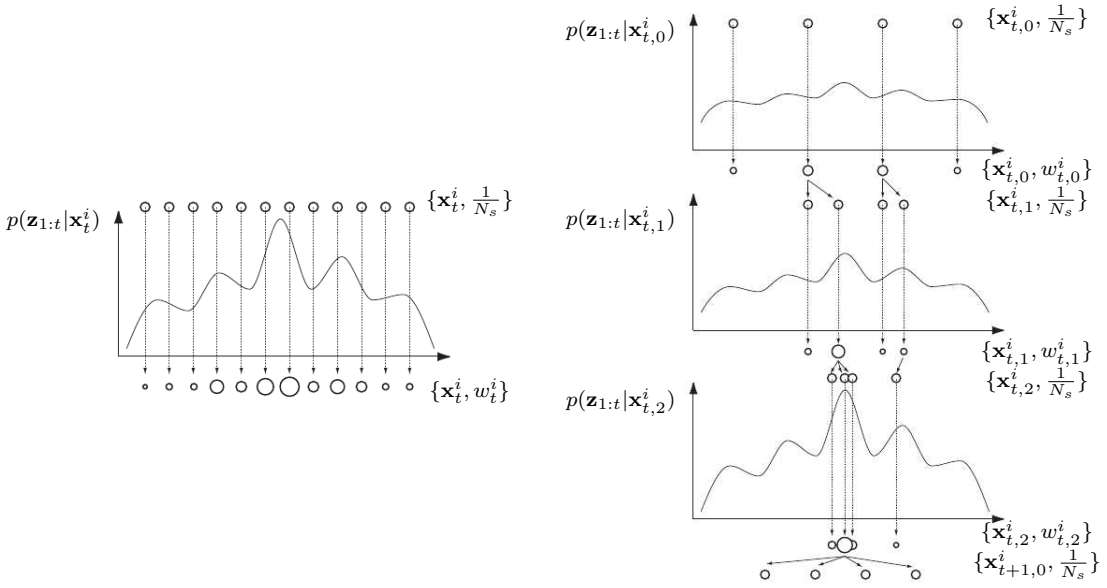
The propagation step can be changed from layer to layer, but usually focuses only on the variables in χ. Throughout this chapter, we will present several propagation strategies rooted on Gaussian diffusion, as well as a data-driven approach that aims at constructing proposal distributions relying on cues obtained from images.

Similarly, we restrict resampling to variables that have been already filtered (see Algorithm 2). By doing so, the sample set is able to retain some diversity on dimensions of the state space for which neither dynamics nor observations have been introduced yet in the filtering process.

Algorithm 2: Layered Resampling

```
Initialize the CDF: cdf_0 = 0
for i = 1 ... N_s do
    cdf_i = cdf_{i-1} + w^i_{t,l}
end
Start at the bottom of the CDF: i = 1
Draw a starting point: u_0 ~ U_{[0, N_s^{-1}]}
for j = 1 ... N_s do
    Move along the CDF: u_j = u_0 + N_s^{-1}(j - 1)
    while u_j < c_i do
        i = i + 1
    end
    Assign variables up to layer l: x^j_{t,l} ← x^i_{t,l}
    Assign weight: w^j_{t,l} = N_s^{-1}
end
```

Layered Particle Filters usually tackle the curse of dimensionality by sampling from a distribution that is a *relaxed* version of the posterior. This usually makes the filter to end up drawing particles only on modes of the posterior with higher probability. As particles concentrate on high probability zones of the posterior, the PF becomes more



(a) (b)

FIGURE 3.3: (a) Particle Filter. (b) Annealed Particle Filter.

suited for optimization, since this phenomenon yields a degenerate estimation of the posterior in multimodal distributions.

3.4.3 Annealed Particle Filter

The Annealed Particle Filter (APF) [DBR00] is a Layered particle Filter that transforms the likelihood function by using concepts of simulated annealing, which is inspired in the behavior of atoms in a metal under the effect of controlled heating and slow cooling. Provided that we have some possibly multimodal and peaky likelihood function $p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)$, the idea is to generate a series of auxiliary likelihood functions $p(\mathbf{z}_{1:t}|\mathbf{x}_{t,l}^i)$ that are smoothed versions of $p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)$. For $l = 0$, the smoothest function is employed, thus emphasizing on global features (low resolution). As l grows, the series of functions slowly converges to $p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)$, thus moving from global features to local features (or higher resolution) (see Fig. 3.3).

The most usual way of obtaining such a series of functions is by setting:

$$p(\mathbf{z}_{1:t}|\mathbf{x}_{t,l}^i) = p(\mathbf{z}_{1:t}|\mathbf{x}_t^i)^{\beta(l)} \tag{3.16}$$

where $0 < \beta(l) \leq 1$ is a monotonic increasing function that determines the annealing rate (as l grows, $\beta(l)$ grows, producing more peaky functions). The series of likelihood functions causes the filter to drive its particles towards higher modes of the posterior (see Fig. 3.3), thus producing a degenerate estimation of the distribution, that makes the APF unreliable in filtering problems. To overcome this limitation, Gall et al. [GPS+07] propose a broader class of APFs termed Interactive Simulated Annealing (ISA) that perform selection of particles during resampling, thus allowing that a fraction of particles with low weights might survive.

There is not an analytical way to find a suitable $\beta(l)$ for a given problem, hence its behavior is set empirically. Deutscher et al. [DBR00] proposed an adaptive method based on the survival rate. They assume that after a layer, a fixed percentage of particles must survive to achieve a proper annealing behavior and thus an adequate concentration of particles near a global maximum. Equivalently, given a target survival rate α_{target}, the following equality must be achieved:

$$\begin{aligned} \alpha(\beta(l)) &= \alpha_{target} \\ \frac{1}{N_s \sum_{i=1}^{N_s} (p(\mathbf{z}_{1:t}|\mathbf{x}_{t,l}^i)^{\beta(l)})^2} &= \alpha_{target} \end{aligned} \tag{3.17}$$

Hence, Deutscher et al. find the actual values of $\alpha(\beta(l))$ by solving Equation 3.17. This equation can be efficiently solved by storing the importance weights and then finding the values of $\beta(l)$ by an iterative optimization algorithm. Although adaptive, this method depends on a likelihood function that will be usually approximated using some cost function. This function might change dramatically from time to time, occasionally causing the adaptation of β to concentrate particles on a very reduced region of the state space. This phenomenon, related to the term over-annealing [CF10], might cause important drifts in tracking. Some other authors have studied the behavior of the APF when using more "controlled" $\beta(l)$ functions. Gall et al. [GPS+07] use several series for $\beta(l)$ (geometric, logarithmic and polynomial) and they conclude that the best behavior for their experiments is achieved by the logarithmic scheme.

The problem of finding a suitable $\beta(l)$ is linked to the problem of setting an adequate propagation scheme for particles. In this thesis, we have adopted from the beginning a Gaussian diffusion scheme, as it is broadly used in the literature. Since annealing starts by exploring the state space globally and ends sampling locally, the variance of the Gaussian diffusion scheme must be in accordance with this behavior, thus must be

progressively reduced as l grows. A good choice for the variances of the multivariate Gaussian used during the diffusion is the following:

$$\Sigma_l = \begin{bmatrix} (\nu^l \sigma_1)^2 & & & 0 \\ & (\nu^l \sigma_2)^2 & & \\ & & \ddots & \\ 0 & & & (\nu^l \sigma_{nx})^2 \end{bmatrix} \tag{3.18}$$

where $\nu < 1$ is a decay rate. The resulting covariance matrix causes particles to concentrate while the APF reaches more detailed layers (higher β). Deutscher et al. intuitively link this decay with the target survival rate of their adaptive annealing scheme: they choose $\alpha_{target} = \nu = 0.5$. However, this choice has limitations, as it depends on the initial variance of each parameter and uses a fixed decay. A scheme termed adaptive Gaussian diffusion, that will be detailed in section 3.5.1, is often employed to overcome the mentioned limitations.

A second look at the APF, from the viewpoint of our *Layered Particle Filter* formalism, reveals that all the variables are introduced in the first layer of the algorithm. One can also see how particles are effectively drawn in APF by looking at Equation 3.14, taking into account that particles are resampled, propagated with Gaussian diffusion and directly evaluated with a smoothed likelihood function: the proposal for every layer is the predictive or prior distribution employed in that layer, but since we have resampled particles coming from the preceding layer, we can understand the APF process as layered filtering with proposal distributions influenced by smoothed versions of the posterior.

Algorithm 3: Annealed Particle Filter

Start in $l = 0$ at time t
Resample particles with replacement $\{\mathbf{x}^i_{t-1,L-1}, w^i_{t-1,L-1}\} \rightarrow \{\mathbf{x}^i_{t,0}, \frac{1}{N_s}\}$
Propagate particles by applying a Gaussian diffusion $\mathbf{x}^i_{t,0} = \mathbf{x}^i_{t,0} + \mathcal{N}(0, \Sigma_0)$
Evaluate particles with the likelihood function $w^i_{t,0} \propto p(\mathbf{z}_t|\mathbf{x}^i_{t,0})^{\beta 0}$
Normalize weights so that $\sum_{i=1}^{N_s} w^i_{t,0} = 1$
for $l = 1, ..., L-1$ **do**
    Resample particles with replacement $\{\mathbf{x}^i_{t,l-1}, w^i_{t,l-1}\} \rightarrow \{\mathbf{x}^i_{t,l}, \frac{1}{N_s}\}$
    Propagate particles by applying a Gaussian diffusion $\mathbf{x}^i_{t,l} = \mathbf{x}^i_{t,l} + \mathcal{N}(0, \Sigma_l)$
    Evaluate particles with the likelihood function $w^i_{t,l} \propto p(\mathbf{z}_t|\mathbf{x}^i_{t,l})^{\beta l}$
    Normalize weights so that $\sum_{i=1}^{N_s} w^i_{t,l} = 1$
end
Compute the posterior mean estimate $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_s} w^i_{t,L-1}\mathbf{x}^i_{t,L-1}$

3.4.4 Hierarchical Particle Filters

Instead of tackling the curse of dimensionality by using concepts of simulated annealing, one can organize the filtering in layers working with a partition of the state vector $\mathbf{x}$. The most generic case of state space partitioning is proposed in Partitioned Sampling [MI00], where $\mathbf{x}$ is split into several subsets of lower dimensionality than the original state space. To successfully deal with the estimation problem, two conditions must be met. First, each subset must have an adequate predictive or prior distribution in order to properly propagate particles. Second, the likelihood function employed in a subset must be peaked around the same region of the posterior restricted to that subset. These two elements ensure efficient propagation and evaluation steps for Partitioned Sampling.

In pose estimation and motion capture, and particularly in analysis-by-synthesis techniques that employ an articulated body model, an implicit hierarchy between variables of the state space exists (see Fig. 3.4. Hence, suitable partitions are naturally found by following this hierarchy. This is the seminal idea of Hierarchical Particle Filters (HPF) that can be considered a particular case of Partitioned Sampling with a hierarchical partition of the state space.

We have introduced pose estimation as the problem of estimating the location of human body joints. In our approach, these joints are part of an articulated model and the positions are defined, according to kinematic chain framework [BM98], by a global rotation, a global translation, joint rotations and bone elongations. Both joint positions and these variables defining them are hierarchically organized in the articulated model: the skeleton is a tree whose root node contains the global variables and the siblings grow defining the kinematic chains or, equivalently, the extremities (arms, legs and neck-head). From this viewpoint, we can realize that the articulated model has the form a Kinematic Tree [BC08] that shows us the natural organization of human body joints in a simple hierarchy. The Kinematic Tree can be used to translate our intuition about the relation between joints and body parts in the human body to probabilistic formulations. Let us consider the model in Fig. 3.4. Let the tree associated to the model be the graph $\mathcal{K}(\mathcal{X}, \mathcal{E})$ where the nodes $\mathcal{X}$ are the joint positions and the edges $\mathcal{E}$ represent the skeleton bones. Now suppose that our problem simply depends on estimating the joint distribution of joint positions, $p(r_o, r_{neck}, r_{head}, r_{right\ shoulder}, \cdots, r_{left\ hip}, r_{left\ knee}, r_{left\ ankle})$ (observations are discarded for the sake of simplicity). In such a graphical formulation, the joint distribution of the set of joint positions can be factorized in a way that the problem is efficiently tackled. For simplicity, let us focus only on the factorization of the joint distribution of the upper body:

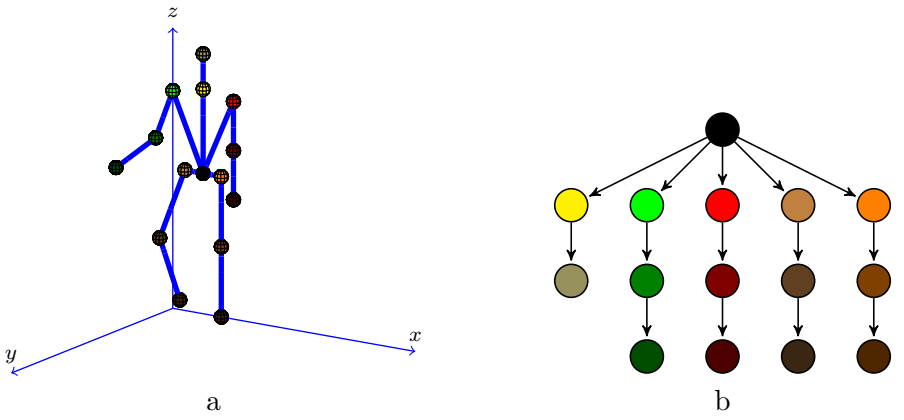


FIGURE 3.4: (a) Articulated 3d model. (b) Equivalent tree representation.

$$p(r_o, r_{neck}, r_{head}, r_{right\ shoulder}, r_{right\ elbow}, r_{right\ wrist}, r_{left\ shoulder}, r_{left\ elbow}, r_{left\ wrist}) = p(r_o)p(r_{neck}|r_o)p(r_{head}|r_{neck})p(r_{right\ shoulder}|r_o)p(r_{right\ elbow}|r_{right\ shoulder}) p(r_{right\ wrist}|r_{right\ elbow})p(r_{left\ shoulder}|r_o)p(r_{left\ elbow}|r_{left\ shoulder})p(r_{left\ wrist}|r_{left\ elbow}) \quad (3.19)$$

Therefore, if the model variables are hierarchically organized according to $\mathcal{K}(\mathcal{X}, \mathcal{E})$, probabilistic graph theory says that an efficient manner of tackling the pose estimation problem is by estimating variables with conditional probabilities such as in equation 3.19. Even if observations are introduced in the model, this factorization can be considered. After all, a benefit from having a graphical model of the body is that a high-dimensional distribution can be compactly represented by means of factorization. In the context of *Layered Particle Filters*, this reasoning can be taken into account by designing subsets $\chi_0, ..., \chi_{L-1}$ following a topological ordering relative to $\mathcal{K}(\mathcal{X}, \mathcal{E})$. However differently from state-of-the-art HPFs, we consider more generic hierarchies by allowing layers not only defined from partitions, i.e, our layered PFs allow subsets with non-empty intersection and thus can perform multiple filtering passes on variables.

Since we initially consider Gaussian diffusion within our *Layered Particle Filtering* framework, a convenient design of the covariance matrix is required. Algorithm 4 presents a method for defining suitable covariances. As in the APF, some decay factor to variables, but in this case, hierarchical subsets are taken into account.

Algorithm 5 summarizes the HPF using these covariances. Since they belong to the class of *Layered Particle Filters*, APF (Algorithm 3) and HPF algorithms are practically identical, except for the likelihood evaluation. One might think that covariances are

Algorithm 4: Compute Σ_l

For new variables in χ_l, $\sigma_n^{'2} = \sigma_n^2$
For any variable included in a subset $\chi_{l'}$, with $l' < l$, $\sigma_n^{'2} = (\nu^{l-l'}\sigma_n)^2$, where $\nu < 1$ is a decay rate.
For any variable included in a subset $\chi_{l'}$, with $l' > l$, $\sigma_n^{'2} = 0$

also different, but they are produced by the same algorithm (in the APF, it suffices with defining a first layer with a subset χ_0 including all the variables in $\mathbf{x}$).

Therefore, having a look at well-know PF reveals the importance of the *Layered Particle Filter* framework. APF and HPF are particular cases of the *Layered Particle Filter*, but not only from a theoretical point of view: it worths implementing a *Layered Particle Filter* because then one can simply change parameters to have an APF or a HPF, or even a mixture between both.

Algorithm 5: Hierarchical Particle Filter

Start in $l = 0$ at time t
Resample particles with replacement $\{\mathbf{x}_{t-1,L-1}^i, w_{t-1,L-1}^i\} \rightarrow \{\mathbf{x}_{t,0}^i, \frac{1}{N_s}\}$
Propagate particles by applying a Gaussian diffusion $\mathbf{x}_{t,0}^i = \mathbf{x}_{t,0}^i + \mathcal{N}(0, \Sigma_0)$
Evaluate particles with the likelihood function $w_{t,0}^i \propto p(\mathbf{z}_t|\mathbf{x}_{t,0}^i)$
Normalize weights so that $\sum_{i=1}^{N_s} w_{t,0}^i = 1$
for $l = 1, ..., L-1$ **do**
    Resample particles with replacement $\{\mathbf{x}_{t,l-1}^i, w_{t,l-1}^i\} \rightarrow \{\mathbf{x}_{t,l}^i, \frac{1}{N_s}\}$
    Propagate particles by applying a Gaussian diffusion $\mathbf{x}_{t,l}^i = \mathbf{x}_{t,l}^i + \mathcal{N}(0, \Sigma_l)$
    Evaluate particles with the likelihood function $w_{t,l}^i \propto p(\mathbf{z}_t|\mathbf{x}_{t,l}^i)$
    Normalize weights so that $\sum_{i=1}^{N_s} w_{t,l}^i = 1$
end
Compute the posterior mean estimate $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_s} w_{t,L-1}^i \mathbf{x}_{t,L-1}^i$

3.4.5 Layered Optimization

Some authors tackle the pose estimation problem by means of local optimization techniques. These techniques might not yield accurate results with commonly used likelihood models, since these models are highly non-linear and multimodal. Due to this multimodality, local optimization techniques usually get stuck in local maxima of the likelihood function. Nonetheless, if a good initial point is provided, local optimization methods can provide accurate estimates. As an example, Gall et al. [GRBS10] use local optimization to refine the estimates provided by Interacting Simulated Annealing (ISA),

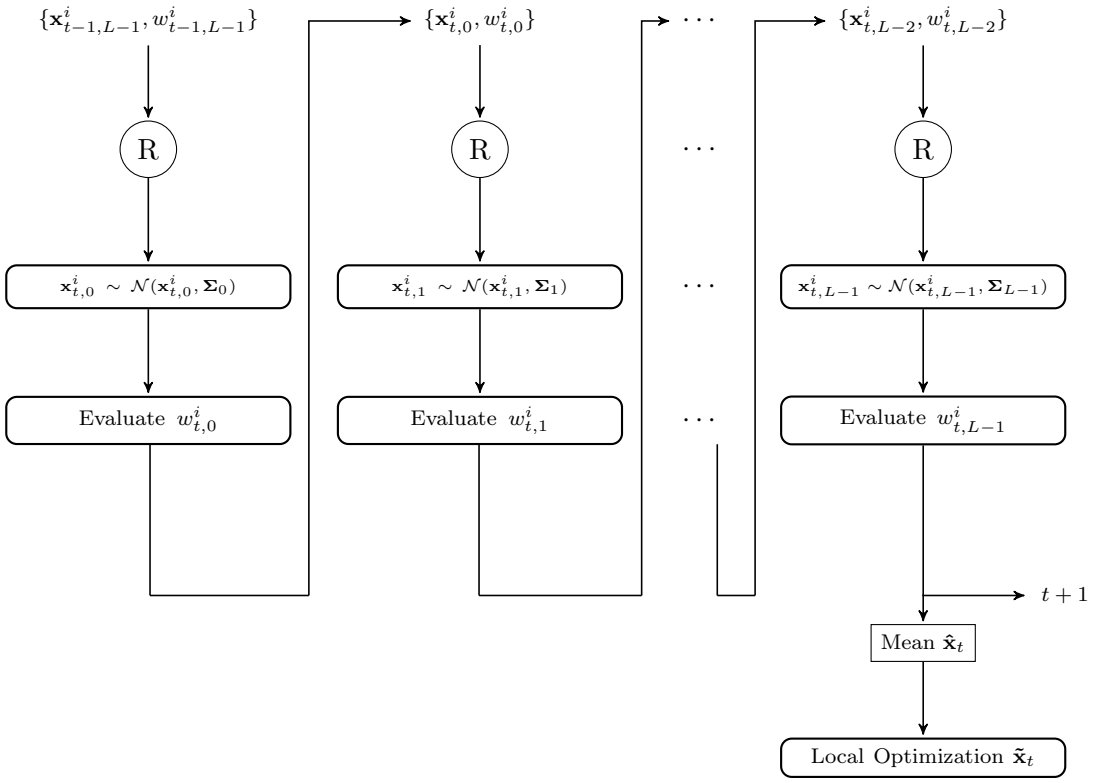


FIGURE 3.5: Layered Optimization Framework as a simple extension of the Layered Particle Filtering scheme.

a generalization of the APF that has the form of a *Layered Particle Filter* with a special resampling mechanism.

The behavior of local optimization techniques with good initial points motivates the inclusion of an optimization step as an additional layer. According to the duality between filtering and optimization (remarked in Section 3.4.1 and in [GRBS10]), one can view the layered particle filtering with the additional local optimization of the posterior mean as a layered optimization method, where in the first layers we rely on stochastic particle-based optimization while the last layer provides local optimization.

In general, local optimization may be considered in a *Layered Particle Filter* by extending the diagram in Fig. 3.2 with local optimization based on the posterior mean estimated by the *Layered Particle Filter* (see Fig. 3.5). For this local optimization step, we employ the method of conjugate directions proposed by Powell [Pow64].

3.5 Improving Propagation

As presented throughout the chapter, our choice for the proposal distribution in every layer is to draw samples from the prior. The simplest form of prior is a Gaussian.

However, this strategy presents a main drawback: particles are drawn blindly with respect to data.

In this section, we present several strategies to overcome the blindness problem to different extents. We start reviewing adaptive diffusion [DR05], a state-of-the-art method for improving the propagation step of APFs. Next, we propose a method to overcome the particle blindness problem: a data-driven layered particle filter with hierarchical layers.

3.5.1 Adaptive Diffusion

Adaptive diffusion [DR05, GRBS10] covers a set of methods where particles are drawn from a distribution with covariance matrix computed from data. We particularize this approach for a multivariate Gaussian distribution whose covariance matrix is the sample covariance matrix:

$$\Sigma = \frac{\alpha}{N_s - 1}\left(\rho\mathbf{I} + \sum_{i=1}^{N_s}(\mathbf{x}_{t,l}^i - \mu_{\mathbf{x}})(\mathbf{x}_{t,l}^i - \mu_{\mathbf{x}})^T)\right) \tag{3.20}$$

where α is a scaling factor, ρ is a small positive scalar that acts as a regularizer and $\mu_{\mathbf{x}}$ is the sample mean. In this way, the amount of drift applied to each variable depends on the particle population. Variables with more uncertainty will be propagated further, while lower uncertainty will cause variables to concentrate around specific values. Due to this selective propagation, adaptive diffusion has been described as a soft partitioning of the state space [DR05].

Although solving to some extent the problems related to manually setting variable covariances or learning them from data containing specific actions, adaptive diffusion depends on the robustness of the observation model and its performance is strongly conditioned by α.

In the following we present an approach that addresses these issues.

3.5.2 Detector-Driven Hierarchical Particle Filters

In the preceding sections, we have presented methods that take into account the particle population in order to draw particles. However, these methods assume that the set of particles in a given layer l carries enough information in order to extract statistics about which variables should be propagated with more uncertainty. This assumption becomes a problem with multimodal distributions, since a number of particles will get stuck on secondary modes (or local minima of the cost function).

In this section, we adopt a different approach. Rather than relying on the particle set, we introduce cues extracted from input images to build an appropriate data-driven proposal distribution that may reduce the blindness of the particle set. Not only that; these cues will, in general, cause the proposal to draw more particles closer to the global minimum or true mode. We achieve this goal by estimating the 3D locations of a set of body parts and incorporating these cues into the *Layered Particle Filter* framework.

In this thesis, we advocate the use of detectors in combination with LPFs with hierarchical layers. On the one hand, we empirically concluded that instantiations of *Layered Particle Filters* containing hierarchically organized layers outperformed APF strategies. This superior performance is especially evident in the estimation of arm and leg poses. On the other hand, having layers where we only estimate one body extremity at a time seems a more convenient way of tackling the pose estimation problem, since the problem of finding poses that match a possibly high number of detections will be split into a hierarchical search of body part poses matching specific detections.

In the literature, body part detectors are widely used in human motion analysis [SNH10], and recently have been proposed as a method for full body pose estimation [SFC+11]. Nonetheless, this last approach requires massive training and, even more important, relies on the fact that range data has much less appearance variability than color and luminance data. Then, it makes sense to point towards a hybrid way of tackling the problem, a mixture between top-down and bottom-up estimation of the human pose.

Throughout this section, when speaking about body parts, we will restrict ourselves to extremities (head, hands and feet). Two main reasons motivate such a restriction. First, it is usually simpler to estimate the location of extremities that other joints, such as knees of elbows. Second, our framework for incorporating body part detections is valid for any body part, but, since it relies on *Inverse Kinematics* (IK), it is actually more effective if we consider only end-effectors or extremities.

In our work, we consider the general case where we have a weakly specialized body part detector, i.e., a detector that can fire positive detections for several body parts. For instance, a skin detector will provide, in general, detections for both hands and head. In this scenario, propagation comprises two main steps: detection assignment and particle generation. Let us consider a LPF where end-effectors (head, hands and feet) are in different layers. In order to determine if any candidate location represents an extremity location, we rely on the pose $\hat{\mathbf{x}}_{t-1}$ estimated on the previous frame. Let F be the forward kinematics operator [HP11] such that $\mathbf{y'}_l = F(\mathbf{x}^i_{t,l})$ gives the joint location of the end-effector associated to l-th layer.

In order to infer the best candidate for each end-effector, we formulate an optimal assignment problem. Let $\mathbf{D} \in \mathbb{R}^{N \times M}$ be the matrix gathering the distances between the N target end-effectors and the M point candidates and let $\boldsymbol{\Upsilon} = \{\Upsilon_1, \ldots, \Upsilon_I\}$ denote the vector of maximum distance assignments (each Υ_n models both the expected movement and the size of a specific end-effector). Assignments are noted as an assignment matrix $\mathbf{P} \in \{0,1\}^{M \times N}$ such that each row contains at most a 1 indicating to which end-effector is the candidate assigned. We consider that an assignment is valid if it exists at least one detected point that satisfies a maximum assignment constraint Υ_i. In that case, the problem has a non-trivial solution ($\mathbf{P} \neq \mathbf{0}$) and is formulated as :

$$\begin{aligned} \min_{\mathbf{P}} \ & \frac{\mathrm{tr}(\mathbf{DP})}{\mathbf{1}_M^T \mathbf{P} \mathbf{1}_N} \\ & \text{s.t.} \quad \mathrm{diag}(\mathbf{DP}) \preceq \boldsymbol{\Upsilon} \\ & \text{s.t.} \quad \mathbf{P} \in \{0,1\}^{M \times N} \text{ is a valid assignment matrix} \end{aligned} \tag{3.21}$$

where $\mathrm{tr}(\mathbf{DP})$ denotes the trace of the matrix resulting of the product between distances and assignments, $\mathrm{diag}(\mathbf{DP})$ denotes the vector formed with the diagonal elements of the same matrix, and $\mathbf{1}_N$ is a vector of ones of length N. The inequality constraint is formulated as a component-wise inequality between two vectors. Hence, we aim at minimizing the overall distance between candidates and end-effectors while maximizing the number of assignments (subject to the maximum distance constraint). In practice, we solve this problem for the human body end-effectors by iteratively assigning pairs with minimum distance until a minimum of Eq. 3.21 is attained. As a result of this assignment, we obtain a maximum of one end-effector detection $\mathbf{g}_{t,l}$ per layer. If $\mathbf{g}_{t,l}$ exists for layer l, then we formulate the distribution of the pose given $\mathbf{g}_{t,l}$ as:

$$\begin{aligned} & \log p(\mathbf{x}_{t,l}^i | \mathbf{g}_{t,l}, \mathbf{x}_{t,l-1}^i) \triangleq \\ & -\frac{1}{2}(\mathbf{g}_{t,l} - F(\mathbf{x}_{t,l}^i))\boldsymbol{\Sigma}_\epsilon^{-1}(\mathbf{g}_{t,l} - F(\mathbf{x}_{t,l}^i)) \\ & -\frac{\lambda}{2}\|\mathbf{x}_{t,l}^i - \mathbf{x}_{t,l-1}^i\|^2 \\ & +\sum_{n=1}^{N} \log \mathcal{U}_{[a_n,b_n]}(\mathbf{x}[n]) + C \end{aligned} \tag{3.22}$$

The first term models the error between an extremity detection and the end-effector position in the model obtained by forward kinematics. The second term is a smoothing constraint, forcing $\mathbf{x}_{t,l}^i$ to be close to $\mathbf{x}_{t,l-1}^i$. The last terms are the kinematic constraints,

formulated as a product of uniform distributions [HP11], and a constant. For brevity, in Equation 3.22 we have omitted the case that expresses this distribution when going from $t-1$ to t. If one minimizes Equation 3.22 (by taking derivatives with respect to $\mathbf{x}_{t,l}^i$), we obtain the pose with maximum probability given a body part location or, equivalently, an IK solution for $\mathbf{x}_{t,l}^i$. Since minimizing Equation 3.22 is equivalent to finding an IK solution and, consequently, a sample with high probability, we can efficiently sample from this distribution. Sampling starts by solving the IK problem with the swing twist formulation [Kal08]. Then, we generate additional samples by random rotations around the swivel axis. This is an efficient way to easily get a number of samples in the typical set of the modes of the distribution in Equation 3.22, because all of them are solutions to the unconstrained IK problem. Furthermore, in this way we generate particles with highly correlated variations between shoulder rotations, which are difficult to generate by simply propagating the corresponding Euler angles. Samples drawn from this distribution are called *IK particles*.

Using this distribution we define a new proposal for our LPF. Since end-effector detections are prone to errors, or might simply be missing, we combine the distribution of Equation 3.22 with Gaussian diffusion, thus obtaining the following proposal distribution:

$$\begin{aligned} q(\mathbf{x}_{t,l}^i|\mathbf{g}_{t,l},\mathbf{x}_{t,l-1}^i) \triangleq \alpha\mathcal{N}(\mathbf{x}_{t,l-1}^i,\mathbf{\Sigma}_{\chi_l}) \\ +(1-\alpha)p(\mathbf{x}_{t,l}^i|\mathbf{g}_{t,l},\mathbf{x}_{t,l-1}^i) \end{aligned} \tag{3.23}$$

where parameter α controls the importance of each mixture component, i.e., whether it is more likely to sample with Gaussian diffusion (regular particles) or IK particles. This parameter considers the error rate of the extremity detections. Instead of estimating such an error rate offline, we propose an online approximation of the detection accuracy by using the *IK survival rate*, i.e., the estimated ratio of IK particles that will be resampled after likelihood evaluation. Whenever this rate is above 0.5, we mutate a small fraction of regular particles into IK particles. On the contrary, when the rate is lower, IK particles are transformed into regular particles. We constrain the algorithm to keep a minimum of 25% of particles of one kind.

In the end, we obtain an approximated method to map 3D end-effector locations into modes of the *Layered Particle Filter* prior distributions in specific layers. These proposals confer our method a hybrid bottom-up and top-down nature. Since the method is based on the re-interpretation of a distribution by means of IK, it is also valid if additional locations (such as elbows or knees) are provided-as mentioned at the beginning of

the section-. However, if additional body part detections are prone to errors, then their incorporation into the proposed framework becomes inefficient.

Note that we have formulated the new proposal considering detectors that provide 3D positions. While in the case of range data it is rather easy to obtain such a position, it is not trivial to detect 3D body parts from multiple views. In the following section, we propose a method to address the problem of body part detection in multiple views.

3.5.3 3D Body Part Detection in Multiple Views

We present a method for 3D localization of body parts from multiple views. In this work, we focus on body extremities (hands, head and feet), as they are usually easier to detect and provide sufficient information to estimate the pose variables related to a kinematic chain. Our method takes advantage of 3D information to deal with occlusions and visibility issues, hence we can robustly fuse the outcomes of one or several image-based detectors working in different views. To achieve such a goal, we first obtain a set of points on the surface of the human body. Second, we compute the probability of each surface point to be an extremity, using the detections on multiple views. Then, the surface points are filtered using a threshold and clustering technique in order to obtain the most likely extremity locations.

The choice of the image-based detectors affects not only the performance, but how the probabilities in each surface point should be treated. We demonstrate and exemplify the method using a simple yet effective image-based skin detector [JR99] (see Fig. 3.5.3). Note that, in any case, more sophisticated body part detectors can be used.

3.5.3.1 Probability Surface

To robustly fuse detections in multiple views, we employ a set of points $\mathbf{q}$ with associated normals $\hat{\mathbf{n}}_q$ lying on the surface of the human body. A suitable set $\mathbf{Q}$ comprising such oriented points can be estimated in a two-step fashion by reconstructing a 3D volume and then computing normals on the volume surface. Alternatively, we opt for the method of Salvador et al. [SSC10] that jointly estimates surface points and normals.

Then, we compute the *visibility factor* $\eta_c(\mathbf{q})$, a value representing the visibility of each oriented point $\mathbf{q}$ with respect to the camera c:

$$\eta_c(\mathbf{q}) = \begin{cases} -\hat{\mathbf{z}}_c \cdot \hat{\mathbf{n}}_{\mathbf{q}}, & \text{if } \mathbf{q} \text{ is visible} \\ 0, & \text{otherwise} \end{cases} \tag{3.24}$$

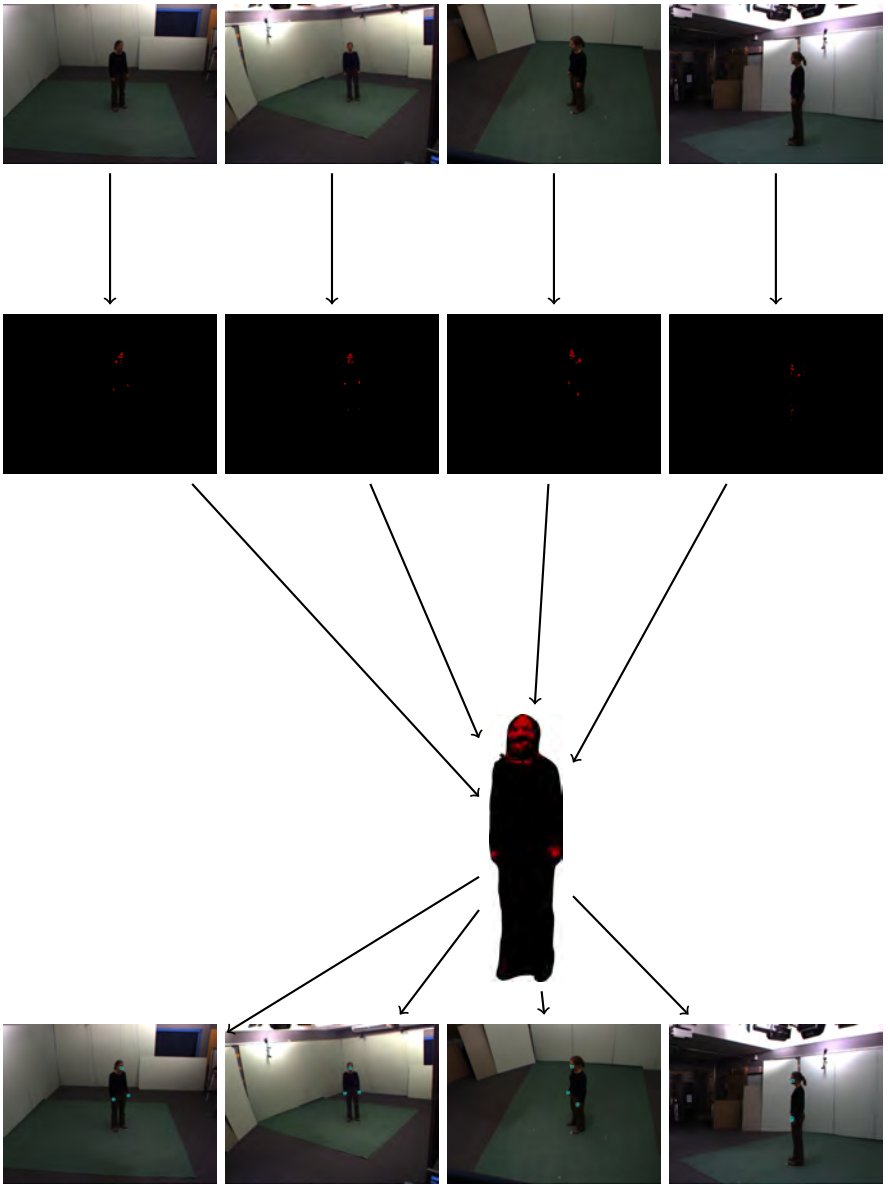


FIGURE 3.6: Mapping 2D detections to 3D world

We determine if $\mathbf{q}$ is visible from camera c by means of a standard z-buffering test.

This visibility factor serves to estimate the probability that a surface point $\mathbf{q}$ is representing an extremity:

$$Prob(\mathbf{q}) = \sum_{c=1}^{C} \eta_c(\mathbf{q}) \mathcal{T}(proj_c(\mathbf{q})) \tag{3.25}$$

where C is the total number of cameras, $proj_c(\mathbf{q}) \in \mathbb{R}^2$ are the pixel coordinates resulting from the projection of the surface point $\mathbf{q}$ in camera c, and $\mathcal{T}(proj_c(\mathbf{q}))$ is the probability that the pixel at $proj_c(\mathbf{q})$ is representing an extremity according to an image-based

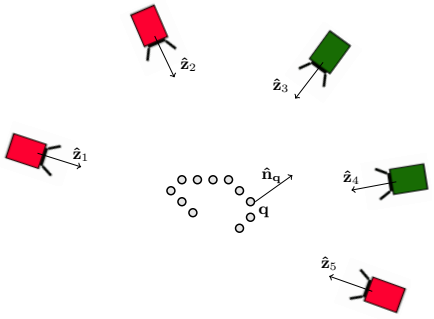


FIGURE 3.7: Visibility example for a surface point $\mathbf{q}$ (best viewed in color). The best oriented cameras are green whereas the red cameras have few or null visibility

detector. Note that the visibility factor has to be normalized, so $\sum_{c=1}^{C} \eta_c(w) = 1$. We show an example in Figure 3.7.

Due to the visibility factor, the proposed method effectively handles occlusions and inconsistencies of the visual cones computed directly from detections in multiple views. Moreover, since only the best oriented cameras determine the probabilities of the surface points, our method can infer a 3D extremity location even if it is reliably detected only in a few views. In addition, as the probabilities of surface points are computed from detections inside the silhouettes, all false positives outside them are not taken into account.

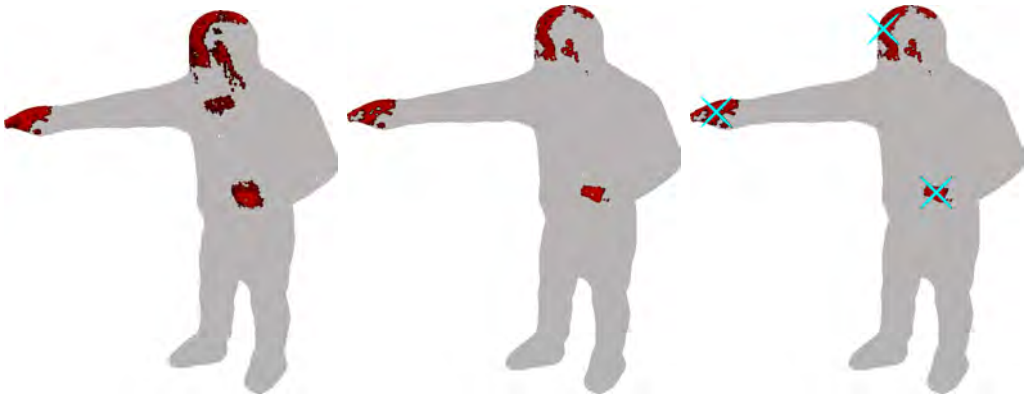
3.5.3.2 Filtering

The filtering step aims at estimating candidate 3D locations of the detected extremities and it is analogous to finding relevant modes of the probability distribution lying on the surface manifold $\mathbf{Q}$. We start by computing the subset $\mathbf{Q}'$ of likely surface points:

$$\mathbf{Q}' = \{\ \mathbf{q} \in \mathbf{Q} \mid Prob(\mathbf{q}) \geq \Gamma\ \} \tag{3.26}$$

where $0 \leq \Gamma \leq 1$ is a threshold.

Then, we cluster the points in $\mathbf{Q}'$ using an efficient method [Rus09] based on a kd-tree representation [FBF77]. The parameters of the clustering method are distance tolerance, ϑ_{tol}, and the minimum and maximum cluster size, ϑ_{min} and ϑ_{max}, respectively. These parameters are very suitable for our problem, since they can be set by using anthropometric proportions. Finally, cluster centroids are taken as candidates for 3D extremity locations. We illustrate the process in Figure 3.8.



(a) (b) (c)

FIGURE 3.8: Filtering process (best viewed in color). Probability values are represented with the red channel (a) Probability surface. (b) Probability surface after the threshold. (c) Cluster centroids (cyan crosses)

3.6 Likelihood Evaluation and Approximate Partitioning of Observations

In computer vision problems, we are not usually able to have access to a real likelihood function. For this reason, the evaluation of the likelihood of a particle is performed by means of an approximation. Let us define a cost function $C_f(\mathbf{x}^i_{t,l}, \mathbf{z}_{1:t})$ that measures the error of rendering the body model according to parameters $\mathbf{x}^i_{t,l}$ with respect to some feature f obtained from images $\mathbf{z}_{1:t}$. Valid and commonly used features f are foreground silhouettes, edges, color, etc.

In this thesis, we do not aim at exploring a great variety of features and how they behave when applied to motion capture. Instead, we investigate the behavior of layered estimators provided simple features that can be obtained from different sensors. In particular, we focus on two features that can be obtained from different sensors. When working with multiple cameras, we use foreground silhouettes. On the contrary, if range cameras are employed, we work with a combination of foreground and depth.

3.6.1 Foreground XOR

Silhouettes in multiple views are commonly used in several pose estimation and body tracking approaches [DBR00, BB09, SBB10]. These silhouettes are usually obtained by background subtraction techniques [Pic04] where one usually builds a color model of the scene when objects of interest are not present. These color models can be pixel-wise [WADP97, SG00] or region-wise [SWFS03, KH09]. A widely adopted method for background subtraction is the Running Gaussian Average [WADP97], an adaptive

pixel-wise Gaussian model in the RGB space. Additionally, shadow suppression methods [XLP05] are introduced to improve the foreground extraction results provided by this technique.

Foreground silhouettes can be compared with the projection of the human body model. Hence, a cost function $C_f(\mathbf{x}_{t,l}, \mathbf{z}_{1:t})$ can be designed for silhouettes. Deutscher et al. [DBR00] initially proposed a cost that only takes into account the number of projected model points that fall inside the silhouette. Given a pose and anthropometric parameters, this cost function measures how well is the model explained given a silhouette, but does not help in measuring to which extent the silhouette is correctly explained by the current model. For this reason, a cost function defined upon the XOR between both the model and the silhouette will have a better behavior, since it takes into account the exact overlap between model and silhouettes. The XOR-based construction of a function that measures the error of projecting the model onto a silhouette is called by some authors as bi-directional cost [SBB10]. Let $\mathcal{S}_v$ be the silhouette in the v-th view and $\mathcal{M}_v$ be the projection of the outer shape of the body model in the corresponding views. Then, the pixel-wise XOR cost between pose and anthropometric parameters $\mathbf{x}^i_{t,l}$ and a set of images $\mathbf{z}_{1:t}$ can be expressed as:

$$C_{fg}(\mathbf{x}^i_{t,l}, \mathbf{z}_{1:t}) = \sum_v \frac{1}{\mathcal{C}(\mathcal{S}_v)} XOR(\mathcal{M}^i_v, \mathcal{S}_v) \tag{3.27}$$

where XOR denotes the number of non-zero pixel values after a pixel-wise XOR, $\mathcal{M}^i_v$ is the projection of the human body model and $\mathcal{C}(\mathcal{S}_v)$ is the number of foreground pixels of the silhouette on the v-th view.

In addition, a pixel-wise cost such as the XOR can be efficiently computed by using a GPU implementation of the XOR. Algorithm 6 provides pseudo-code detailing how an XOR can be computed in parallel for many pixels and views using OpenGL routines.

3.6.2 Depth SSD

In recent years, range sensors have proliferated to an extent that the computer vision community has produced a notable number of contributions that are based on depth data. Among the most remarkable areas of application, markerless motion capture stands on its own, since it becomes more feasible to implement methods for pose estimation with robustness against lighting conditions, appearance changes and self-occlusions. Most of these methods usually rely on a cost function that aims at measuring how well a pose fits on the observed data. These data could be whether the depth map [SM10]

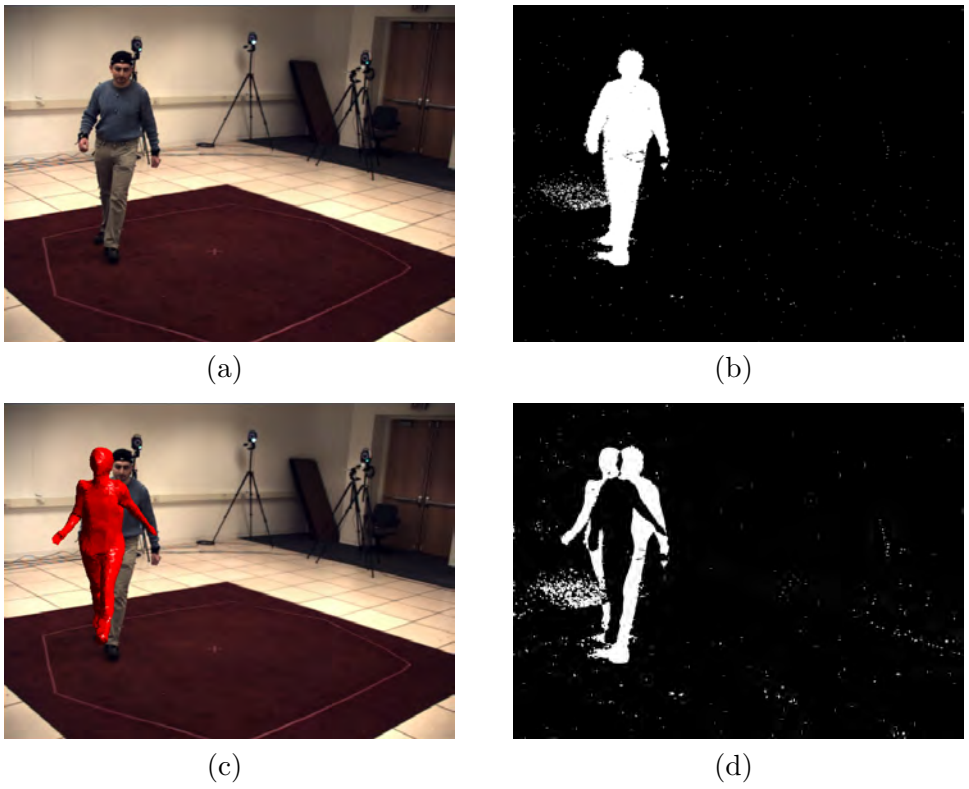


FIGURE 3.9: XOR Cost function. (a) Input image. (b) Foreground Silhouette (a). (c) Body model (overlaid on the image). (d) XOR result.

Algorithm 6: Silhouette XOR

```
1 Load silhouettes as GPU textures using alpha channel.;
2 foreach camera c do
3     Enable writing only in the bit plane c of the stencil buffer.;
4     Set the stencil test to pass always. The stencil operation is set to replace the buffer
      values by 1 on the bit plane c.;
5     Set the camera projection matrix with the parameters of camera c. Render the
      model.;
6     Enable the alpha test. Set the stencil test such that it is passed when the buffer bit
      value equals 1. Set the stencil operation such that it replaces the stencil bit value
      by 0 when it is 1.;
7     Set an orthographic projection matrix. Bind the texture associated to camera c
      and render a quadrilateral polygon of image size.;
8 end
9 Transfer the stencil buffer from GPU to CPU.;
```

or a 3D point cloud [LAaKR10]. Depth-based cost functions are defined as the distance between data points to the closest body model points, in a sort of Hausdorff metric [HKKR93]. This usually implies finding a nearest neighbor and then computing distances between points.

Our choice for the cost function with range data is to use the depth map. This choice

is motivated by the implementation efficiency. First, the problem is tackled in a lower-dimensional search space, since we can work in 2-dimensional image coordinates. Second, we can take advantage of image coordinates also when trying to assign nearest neighbors: we simply consider that projected model pixels must match with data pixel locations, thus the assignment is straightforward (we omit nearest neighbor search). Finally, we consider a GPU design of the cost function, and using images (even of they are of floating point values) is usually easier and less expensive in terms of memory usage than using 3D point clouds.

The cost function is defined in terms of a Sum of Squared Differences (SSD) between data and the model. To do so, we render compute the body model configuration after setting the pose and anthropometric parameters $\mathbf{x}_{t,l}^{i}$ and we render this model using Z-buffering methods (OpenGL depth buffer). This rendering step is performed with the same camera parameters of the range sensor, so we end up having a synthetic depth image $\mathcal{M}_d$ produced by our body model that has the same image support U than the data depth map $\mathcal{D}$:

$$C_d(\mathbf{x}_{t,l}^{i}, \mathbf{z}_{1:t}) = \frac{1}{U}\sum_{u}^{U} \|\mathcal{D}(u) - \mathcal{M}_d(u)\|^2 \tag{3.28}$$

Although its use is not shown in this chapter, we refer to this cost function later in this section. For a more in-depth application of this cost function in our layered framework, see Appendix B.

3.6.3 Collision

A very important issue when working with human body models aiming at pose estimation is to avoid interpenetration of limbs. The interpenetration avoidance, together with the hard kinematic constraints defined in the skeletal model, implies a reduction of the search space so that the pose estimation problem can be tackled more efficiently.

Setting interpenetration restrictions is equivalent to quantifying collision depth. This is easier to handle if we define a set of spheres approximating the shape of the different limbs and then compute the distances between sphere centers.

These spheres reduce the problem of quantifying the collision depth to computing the distances between sphere centers. If we define a set of pairs of potentially colliding limbs L, the evaluation of the collision depth ζ is approximated as follows:

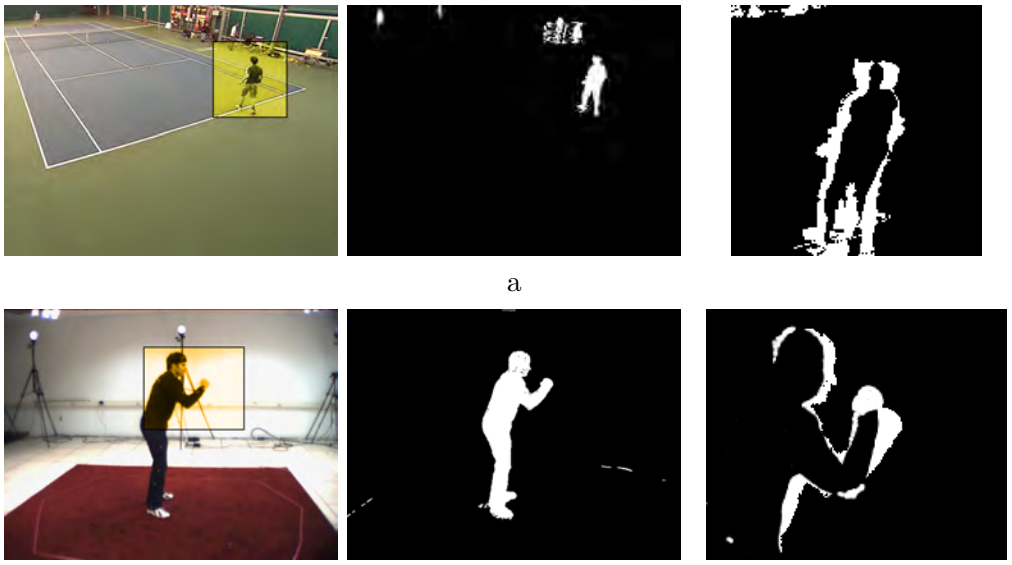


FIGURE 3.10: Examples where only a small region of the images may be relevant for specific state-space variables. **a**: Only a small region of the image is representing the target player. Overlapping the model projection onto the pixels enclosed by the bounding box yields error masks like the one shown on the rightmost, thus avoiding severe errors due to false positives. **b**: Pixels in the marked region are expected to be relevant for estimating the right arm pose. Note that errors due to leg configurations will not affect if pixels on the rightmost mask are used.

$$\zeta = \sum_{\{i,j\}\in L} \sum_{k\in Sph_i} \sum_{l\in Sph_j} d_{k,l} u(d_{k,l} - r_{k,l}) \tag{3.29}$$

$$d_{k,l} = \|\mathbf{c}_k - \mathbf{c}_l\| \tag{3.30}$$

$$r_{k,l} = r_k + r_l \tag{3.31}$$

where $\{i, j\}$ are the indices of the potentially colliding limbs, Sph_i denotes the set of spheres approximating the shape of the i-th limb (k and l are use to denote the respective spheres within these sets), $u()$ is the Heaviside function and r_k and $\mathbf{c}_k$ are the radius and the center of the k-th sphere.

3.6.4 Approximate Partitioning of Observations

Let us recall that in section 3.4.4, we have presented the sampling strategy of standard HPFs as a particular case of Partitioned Sampling. Even in the more general formulation of LPFs, any sampling procedure dealing with subspaces of the state-space (such as Partitioned Sampling) requires a set of likelihood functions, one per subset, that must be properly peaked around the same region of the posterior restricted to the subset

[MI00]. In practice, this requirement is difficult to fulfill with observation models based on silhouettes or depth due to several reasons. First, in a wide base-line multi-camera scenario, the appearance of humans in distinct views may change considerably, causing the ratio between the number of pixels representing the human shape and the total number of pixels in an image to change drastically between views. Furthermore, this might be a reason for possible spurious detections of foreground objects to have more impact on the likelihood function (Fig. 3.10a). In the case of human body, silhouettes and depth are rather incomplete as evidence; it is difficult to ensure that a body part is being matched with a set of pixels that constitute the actual representation (or a very good approximation) of that body part (Fig. 3.10b). This phenomenon is especially remarkable when self-occlusions between body parts occur.

In existing HPFs[BB09], a common solution is to render the body parts that are affected by variables being sampled in a subset or hierarchical level and the preceding ones. Our proposal is to go one step further by defining approximate partitions of the silhouettes or depth data. Our approach is inspired by the windowing proposed in [TCMS03] for parallel optimization architectures aiming at pose estimation. Differently from [TCMS03], we aim not only at a faster but at more accurate likelihood evaluation. Furthermore, our approach is applied to particle filtering methods.

The goal is to restrict the cost function to image regions that are likely to represent the body parts being affected by variables in a given hierarchical layer. By doing so, we reduce the impact of pixels that are not providing meaningful information about variables in the hierarchical layer of interest. For that purpose, we define a set of binary hidden variables Ψ associated to each pixel in every available view. If a hidden variable takes value 1, the associated pixel is taken into account for the computation of the cost function $C_f(\mathbf{x}_{t,l}, \mathbf{z}_{1:t})$. Otherwise, the pixel is discarded. Due to their binary nature, hidden variables define partitions on the image domain. We use these hidden variables to restrict the pixel domain in the cost functions $C_f(\mathbf{x}_{t,l}, \mathbf{z}_{1:t})$ to a set of pixels with an expected relevance for the body part associated to variables in the l-th subset. In the following, we explain how these variables are introduced in the likelihood model, and how are they computed.

Let us recall that we have a human body model that provides us the necessary information to compute joint positions from state-space variables, and that we have also a set of projective transformations p_v, given by the camera calibration, that map points in $\mathbb{R}^3$ onto each view v. We illustrate the concept with a vector formulation of the XOR cost function (see Equation 3.27). Let us express foreground images in multiple views as the vectors $\mathbf{Z}_v \in \{0,1\}^{W_v H_v}$, i.e., foreground images of width W_v and height H_v are gathered in a set of vectors. We drop the layer and time indices for simplicity. Given

particle $\mathbf{x}_{t,l}^i$, projective transformations p_v and a body model, we can draw the vectors $\mathbf{X}_v \in \{0,1\}^{W_v H_v}$, i.e., we project the body model onto multiple views and we construct a vector out of the obtained images. Then, the cost function in Equation 3.27 can be expressed as:

$$C_{fg}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) = \sum_v \frac{\mathbf{1}^T |\mathbf{Z}_v - \mathbf{X}_v|}{\mathbf{1}^T \mathbf{Z}_v} \tag{3.32}$$

where $\mathbf{1}$ is a vector of ones of length $W_v H_v$. This vector reflects the use of all the pixels in $\mathbf{Z}$ in order to compute the likelihood of a pose. Our approach substitutes this vector by a more informative set of hidden variables, gathered in a vector $\psi_{v,l,t}$. For the sake of readability, we drop the time and layer subindices of the hidden variables and we re-formulate the cost:

$$C_{fg}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) = \sum_v \frac{\psi_v^T |\mathbf{Z}_v - \mathbf{X}_v|}{\psi_v^T \mathbf{Z}_v} \tag{3.33}$$

Now the question is how to compute the hidden variables. We discard learning them from data, as we should learn them for many different views and poses, thus rendering a huge learning problem (probably prone to overfitting). Instead, we adopt a simple yet efficient spatial criterion. Since we know the projective transformation and we have a body model, we can compute how far is a pixel from any projected joint. Consequently, we can define partitions based on the proximity to body parts that are being filtered in a layer.

For a given hierarchical subset l, we have a set of particles $\mathbf{x}_{t,l}^i$ and associated importance weights $w_{t,l}^i$. Using this information and the fact that we have already filtered the parent variables (variables for subsets 0 to $l-1$), we compute hidden variables Ψ_l as follows:

1. Compute the sample mean $\hat{\mathbf{x}}_{t,l-1} = \sum_{i=1}^{N_s} w_{t,l-1}^i \mathbf{x}_{t,l-1}^i$ and obtain the 3D position of the joints in the l-th subset, say $\mathcal{Y}_l$.

2. Compute a 3D Bounding Box centered at the mean joint location $\mu_{\mathcal{Y}_l}$ and with fixed sizes proportional to the estimated anthropometric sizes of parts involved in the l-th subset. Compute a 2D Bounding Box on each image enclosing the projected 3D Bounding Box corners.

3. Set $\psi = 1$ for pixels inside the 2D Bounding Box and $\psi = 0$ otherwise.

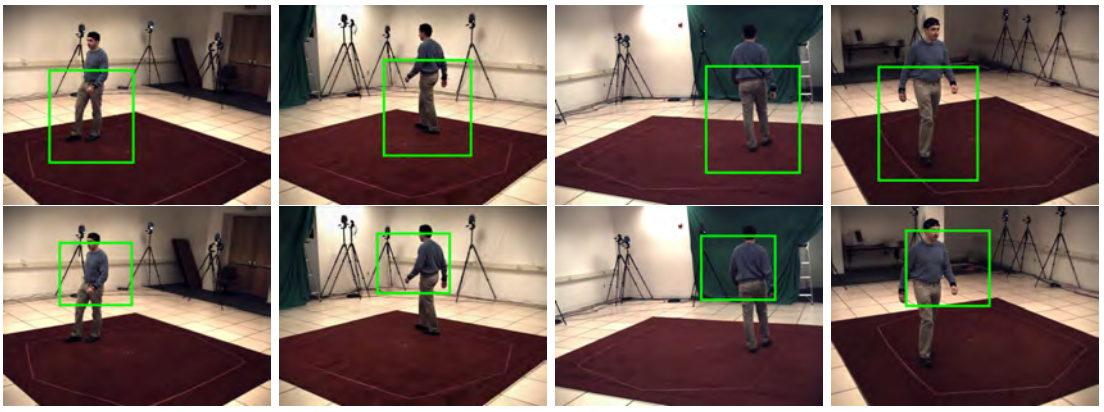


FIGURE 3.11: Examples of the approximate projection of bounding boxes for different layers. Top row: Bounding boxes for legs' layers. Bottom row: Bounding boxes for left arm layers

Restricting these partitions to have the form of bounding boxes in every view makes the algorithm computationally simple, suitable for GPU processing and allows avoiding explicit 3D reconstructions of the data.

The proposed partition can be computed for range data. Equations 3.32 and 3.32 can be reformulated to express the SSD cost function of Eq. 3.28.

Note that the proposed partitioning method does not solve the problem of self-occlusions between body parts, but in some sense, alleviates it. Suppose that the arm is being occluded in one view by the torso. One must take into account that, when using silhouettes, views behaving in such a way are not informative and should have a low impact on the likelihood measurement. Since the proposed partitioning will often manage to block errors originated by unrelated body parts, such as legs or the other arm, the final likelihood will be, in general, better defined in the presence of occlusions.

An important consideration of the proposed likelihood measurement is the fact that particle weights may reflect the posterior distribution of a subset χ_l rather than the whole state-space. This is due to the local nature of the measurement on the partition domain. To overcome this issue, we add an additional layer at the end of the HPF algorithm, in which we simply re-evaluate the weights of the particles using a bounding box that encloses the whole body model.

3.6.5 Refinement Layer

In the preceding section, we have noted that an additional layer evaluating the whole state vector helps in finding a pose estimate (through posterior mean) with less estimation error. In case we are using body part detectors for enhancing the propagation step

3.5.2, we can utilize these detections to refine the final pose estimation. For that matter, a set of extremity detections are introduced in the evaluation step of an additional layer. We use the whole set of available detections in order to detect possible misassignments occurred after solving the assignment problem in Equation 3.21. We denote this whole set of detections as $\tilde{\mathbf{g}}_t = \{\gamma_t^1, .., \gamma_t^u, ..., \gamma_t^U\}$. We add a new cost function taking into account the distance between detections and the model end-effectors. This cost is computed as:

$$C_{dd}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) = \frac{1}{U} \sum_{u=1}^{U} \min_k \parallel \gamma_t^u - F_k(\mathbf{x}_{t,l}^i) \parallel \tag{3.34}$$

where F_k is the forward kinematic operator on the k-th kinematic chain of the articulated model. The objective of this cost is to measure how well matched detections $\mathbf{g}_t$ are explained by the pose $\mathbf{x}_{t,l}^i$. This cost is combined with the cost function of Equation 3.27:

$$C_{rf}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) = C_{fg}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) + \kappa C_{dd}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t}) \tag{3.35}$$

where $C_{fg}(\mathbf{x}_{t,l}^i, \mathbf{z}_{1:t})$ is the foreground cost evaluating the whole silhouette and κ is a scaling factor. κ is chosen to balance the importance of both cost terms. In general, adding the cost C_{dd} in the likelihood approximation of a refinement layer provides more accurate results in terms of end-effector positioning. Furthermore, it has the ability to filter particles representing poses with mismatched end-effector positions.

The refinement layer implies a re-weighting of all the particles according to an improved observation model, and does not involve any propagation step, since we do not want to add noise to particles that have been drawn after layered filtering. If no detections are found at time instant t, κ is set to 0.

3.7 Experimental Results

In order to objectively evaluate the performance of the proposed framework, we conduct experiments on several publicly available datasets. In all these datasets, we employ a skeletal model with 23 DOF and a deformable mesh model obtained by meshing a visual hull reconstruction. The mesh is attached to the skeletal model by means of automatic rigging [BP07]. Since we do not aim at evaluating the impact of the modeling framework, but to compare the estimators, we use a single model in all the trials. However, in order to adapt the same model to different subjects, we automatically

estimate suitable anthropometric parameters at the beginning of each sequence. We perform such a estimation by using the same LPF framework presented in this thesis, but applied to anthropometric variables. This work, involving automatic estimation of anthropometric parameters of skeletal models, has been developed in more depth by Marcel Alcoverro in [ACP10].

3.7.1 HumanEva

We evaluate the Layered Particle Filtering framework with several experiments on the HumanEva II dataset [SBB10]. This dataset contains two sequences recorded by 4 calibrated cameras with resolution 640x490 at 60 fps. The two sequences belong to two different subjects, namely S2 and S4. The ground truth has been obtained with a marker-based motion capture system and it is synchronized with the video data. Using the online evaluation system [SBB10], we measure the absolute 3D error of the estimated joint positions and we report the mean and standard deviation over all the computed frames. As stated by the dataset authors [SBB10], frames 298-335 in S4 sequence are corrupted, and hence are not used for reporting results.

Since markers for motion capture data are placed on the subjects, and we estimate joint positions as bone joints, we register marker positions on the mesh surface at the beginning of each sequence (see Fig. 3.12). Specifically, we employ frame 2 of both sequences in order to perform this registration (marker positions are available for a few initial frames). By doing so, we reduce the constant error caused by the actual distance between joint positions and marker positions. This is only a partial reduction since after registration the error is 4 cm approximately for both subjects. In addition, by registering the markers to mesh vertices, we slightly increase the standard deviation of the error. In spite of these registration errors, the overall results are better with the registration process.

In our first experiment, we start by evaluating the performance of using hierarchical layers versus the APF. To test the annealing-based strategies, we instantiate the LPF with an annealed particle filter using Gaussian diffusion (APF) and using adaptive diffusion (APF-AD). Additionally, following the layered optimization approach, we perform a local optimization step in conjunction with the APF-AD (APF-AD+LocalOpt). For the hierarchical strategy, we instantiate our LPF with 7 hierarchical layers defined as follows:

- **Layer 0** Global translation and rotation
- **Layer 1** Upper legs

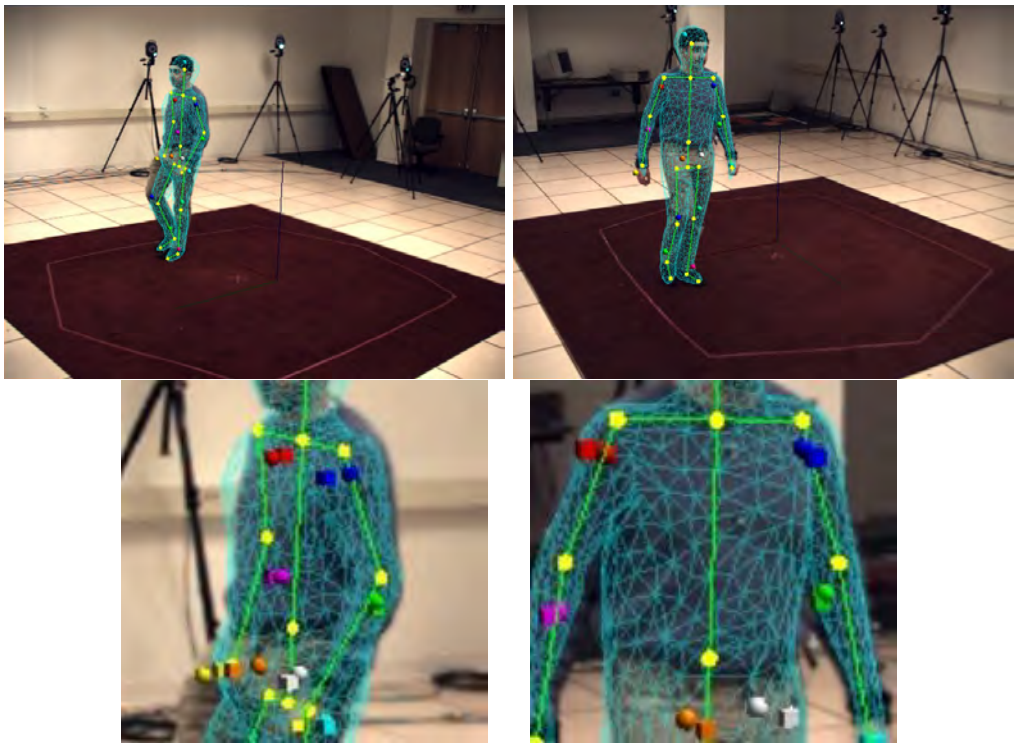


FIGURE 3.12: Marker registration process (best viewed in color). Top: Ground truth markers represented by colored spheres. Bottom: Detail of the registered markers. A registered marker is the vertex with minimum distance to a given ground truth marker. These vertices are drawn with a colored cube.

- **Layer 2** Lower legs
- **Layer 3** Upper right arm
- **Layer 4** Upper left arm
- **Layer 5** Lower right arm
- **Layer 6** Lower left arm

We use these layers for layered filtering (HPF) or with an additional local optimization step (HPF+LocalOpt). We also evaluate the Approximate Partitioning of Observations (APO) with this hierarchical configuration of layers (APO-HPF) and within the layered optimization framework (APO-HPF+LocalOpt). Notice that, compared to existing hierarchical sampling and hierarchical particle filtering strategies, the proposed APO-HPF includes a new layer where the likelihood function given the complete state vector is evaluated.

For the HPFs, we use 500 particles per layer. On the other hand, as it is reported that simulated annealing benefits from more and finer iterations [GRBS10] (or layers for APFs), we employ APFs with 14 layers and 250 particles per layer. Finally, we use 437

Method	S2 Error (mm)	S4 Error (mm)	Time (sec./frame)
APF	99.2 ± 22.1	104.4 ± 25.7	20
APF-AD	78.3 ± 15.3	78.5 ± 23.2	21
APF-AD + Local Opt.	81.6 ± 25.7	83.5 ± 29.2	37
HPF	75.6 ± 12.8	74.5 ± 19.3	21
HPF + Local Opt.	69.6 ± 10.7	58.7 ± 15.4	37
APO-HPF	74.7 ± 12.8	72.5 ± 14.1	14
APO-HPF + Local Opt.	71.4 ± 8.9	61.7 ± 8.3	30

TABLE 3.1: Pose inference results on the HumanEva-II dataset.

particles per layer in the APO-HPF setups, as it uses an extra layer in order to evaluate the global likelihood. Notice that in all the cases we perform a total of 3500 evaluations of the likelihood function.

Mean and standard deviation of the 3D error are provided in Table 3.1. We also report the computational efficiency of the algorithms in seconds per frame. The computational time has been obtained in an Intel 2.40GHz CPU with 4GB RAM. The employed machine is equipped with an NVIDIA GeForce GTX 295 GPU. In spite of the available graphics hardware, our implementation is a single threaded C++ program with some OpenGL routines needed to render the mesh and to compute the cost function as described in Section 3.6.1.

The obtained results show that LPFs with hierarchical layers consistently outperform annealing-based strategies. Probably, the most significant figure is the performance obtained with local optimization in combination with the APF. A reported, the use of local optimization with APFs not only does not improve the results, but delivers a worse accuracy. This result indicates that annealing strategies do not efficiently explore the state-space. From an estimation viewpoint, this means that the APF delivers a highly degenerated posterior. Likewise, from an optimization viewpoint, annealing does not succeed in providing a global optimum of the cost function employed in this dissertation. Contrarily, local optimization is highly beneficial when used with hierarchical layers, which provide accurate initialization poses for local optimization.

Although expected, we experimentally verified that adaptive diffusion improves the accuracy of the APF. However, differently from [GRBS10], we obtained a better performance by setting $\alpha = 0.2$. Although we employ a different observation model, solely based on silhouettes, we also noted that if α is not properly set, the performance of adaptive diffusion dramatically decays.

The APO-HPF does not provide a substantial gain in accuracy, in fact the mean error is slightly higher with respect to the HPF when using local optimization. Because there is some bias when registering the markers to the ground truth, we argue that standard

Method	S2 error	S4 error	Time	Frames
[SBB10]	83 ± 7	78 ± 15 [1]	250	Full
[HWG11]	173	144	10	Full
[GRBS10]	38 ± 9	32 ± 5	124	Full
[CFCP11]	58 ± 31	62 ± 34	-	Full
[BJB12]	-	58 ± 13	11 [2]	Full
[PVW10]	107 ± 8	92 ± 21	36	-380
[CMG+10]	78 ± 10	80 ± 13	-	-150
[RRR09]	75	82	-	Full
[YGvG12]	42 ± 9	42 ± 13	4	Full
APF-AD	78 ± 15	79 ± 23	21	Full
HPF+LocalOpt.	70 ± 11	59 ± 15	37	Full
APO-HPF+LocalOpt.	71 ± 9	62 ± 8	30	Full

TABLE 3.2: Comparison to the state-of-the-art for HumanEva II. The methods are often not directly comparable since they rely on different assumptions. Different body models are used, some methods rely on motion models and some even stress the robustness against low frame-rates. Mean 3D error (in mm), standard deviation (in mm) and computational time (in seconds per frame) are provided (whenever reported by authors in their original works). If results are not provided for the full S2 and S4 sequences, we provide a comment on the column *Frames*. All the provided methods employ 4 cameras in order to infer the human pose.

deviation becomes a highly important indicator of the consistency of the tracking algorithm (whenever the mean 3D error is sufficiently low). In that sense, and compared to the rest of approaches, combining APO-HPF with Local Optimization delivers the lower standard deviation while keeping a sufficiently low mean 3D error. Later in this section, when comparing our results to the state-of-the-art, we discuss the 3D error bias issue in more depth.

Apart from the presented quantitative results, Figs. 3.13, 3.14, 3.15, 3.16, 3.17 and 3.18 provide some qualitative results for different approaches and both subjects S2 and S4.

Regarding the computational efficiency of the presented approaches, we found that in the HumanEva-II dataset, using the approximate partitioning of observations as proposed does not yield a much faster method, since the projection of the bounding boxes is rather big in all the views. However, in other setups, we have experienced a much higher gain (up to twice the speed) and with the 2D bounding boxes we can compute more likelihood evaluations in parallel with the same GPU memory (see Appendix B for examples with range data).

In Table 3.2, we provide a comparison with state-of-the-art methods that reported results on the HumanEva II dataset. Our approximate partitioning of observations in LPFs with hierarchical layers compares favorably to the majority of state-of-the-art approaches both in accuracy and computational performance.

The ISA approach by Gall et al. [GRBS10] and the variation using action priors [YGvG12] are the only approaches that deliver a substantially better performance in

terms of mean of the 3D error. However, [GRBS10] is much slower than our approach (more than 3 times slower) and [YGvG12] relies on strong priors obtained by motion models of actions embedded in specific manifolds. Furthermore, both approaches are based on the ISA framework, which is, in essence, a simulated annealing-based approach to drive particles towards the global minimum of the cost function. Hence, these approaches could benefit from our APO with hierarchical layers. We also observe two competing approaches. On the one hand, [BJB12] delivers a similar accuracy with increased computational efficiency thanks to self-learned motion models. Interestingly, this approach has a similar philosophy based on hierarchical partitions combined with annealed particle filters. On the other hand, the scalable models proposed in [CFCP11] are reported to provide a competitive accuracy in terms of mean error. However, standard deviation associated to these errors is much larger than in our approach (> 30mm). This deviation matches some important errors observed by visual inspection of the tracking results [3]. Visually, our approach delivers a better performance. This last observation around means and standard deviations of the error in the HumanEva II dataset yields to a discussion previously raised in [BJB12]. We observe that the error rarely goes below 5 cm during all the sequence (except for the first frame, where the markers have been registered). This actually means that the 3D errors is not without bias. This bias can be caused by measurement errors of the mocap and by the impossibility to perfectly locate the ground truth markers on the estimated model. In such case, the standard deviation becomes a very good indicator of the quality of the tracking. Observe that [SBB10], [GRBS10], [YGvG12], [BJB12] and our hierarchical approaches deliver not only lower mean error than other state-of-the-art methods, but also a substantially lower standard deviation. But most importantly, note that only [GRBS10] and our APO-HPF with local optimization approach deliver a consistently and substantially lower deviation in both sequences. In particular, we are able to deliver deviations below 1 cm, which indicates that our method consistently tracks the subject's limbs throughout the sequence.

3.7.2 DD-HPF in IXMAS

Since most of the presented filtering strategies succeed in tracking arms and legs in the HumanEva II dataset, we conduct experiments on the IXMAS dataset [Inr06]. The aim of these experiments is to provide a better evaluation of the data-driven strategy presented in this thesis. The IXMAS dataset was recorded for action recognition and hence it does not contain pose ground truth. For this reason, we manually annotate

[1] Average over the results provided in [SBB10]

[2] Average between the full and low-variance mode, considering the 40% of time the algorithm can be accelerated, according to the authors.

[3] Visual results are reported in the online version of [CFCP11]

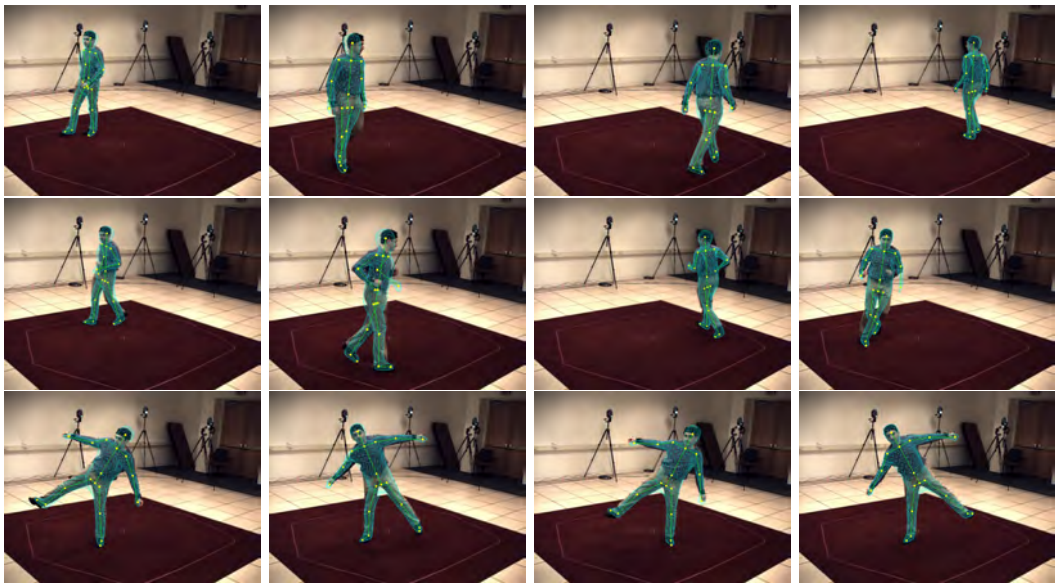


FIGURE 3.13: APF + Adaptive Diffusion + Local Optimization results for Subject S2 on the HumanEva II dataset.

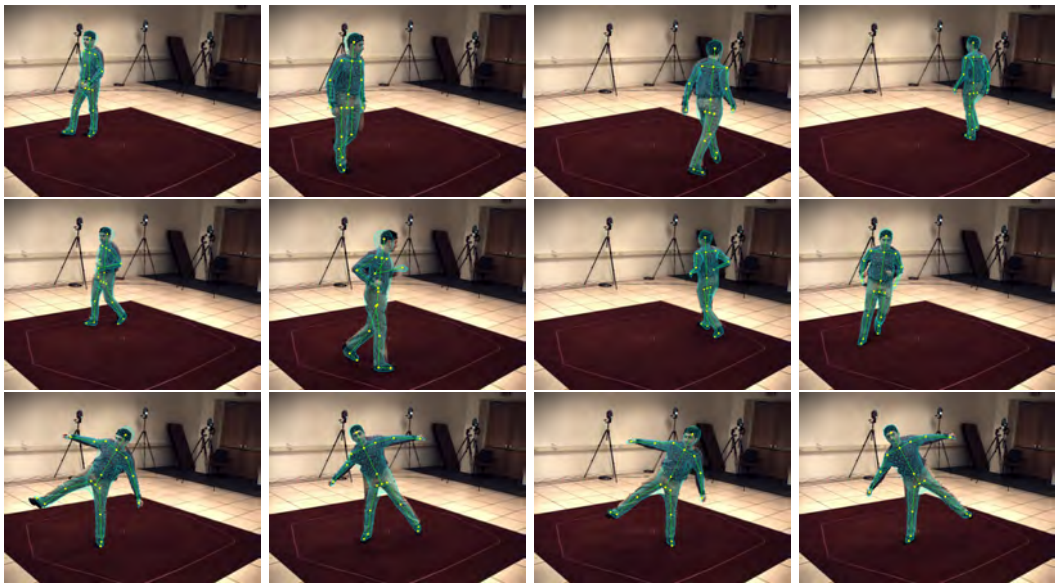


FIGURE 3.14: HPF + Local Optimization results for Subject S2 on the HumanEva II dataset.

the hands, head and feet of different sequences belonging to 6 different subjects. In particular, we are interested in the evaluation of the performance of our method for actions involving arm movements (i.e crossing arms, hand waving, punching, etc.), so we skip actions not involving relevant arms motion such as walking or turning. Annotations are performed in 1 of every 5 frames[3].

We compare our detector driven method with two state-of-the-art *Layered Particle Filters*: the APF and the HPF. In particular, we evaluate our method with (DD HPF$^+$)

[3]For annotations and related data check *https://imatge.upc.edu/∼marcel/ddhpf.html*

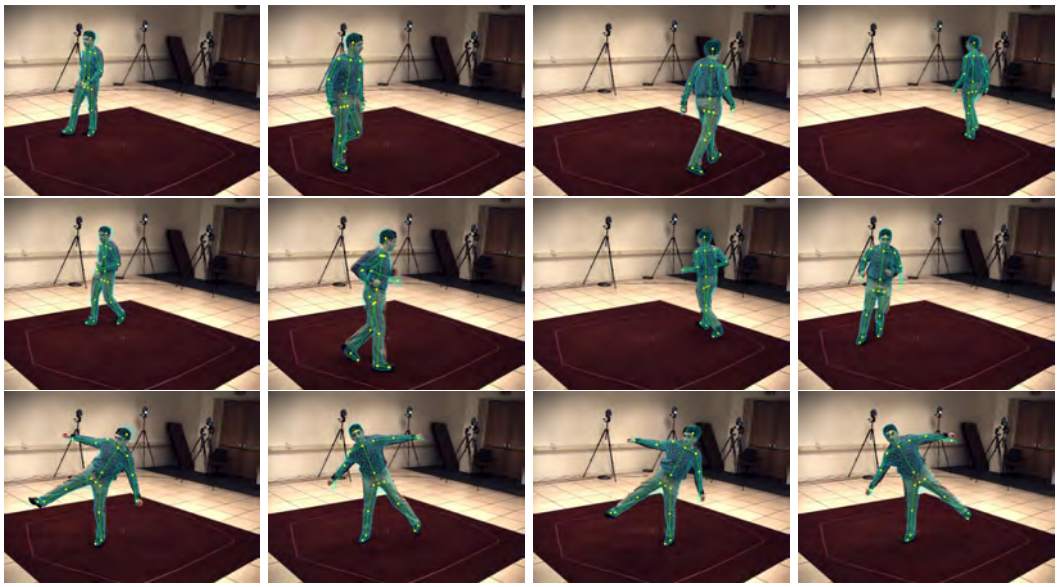


FIGURE 3.15: APO-HPF results for Subject S2 on the HumanEva II dataset.

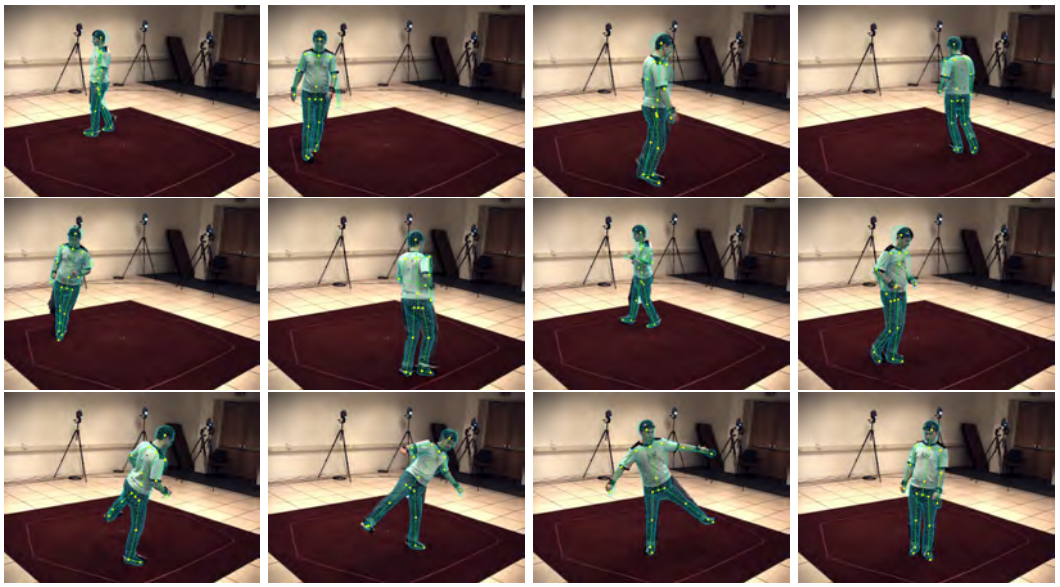


FIGURE 3.16: APF + Adaptive Diffusion + Local Optimization results for Subject S4 on the HumanEva II dataset.

and without (DD HPF) the refinement layer. In order to perform the comparative, the APF is run with 14 layers and 250 particles per layer, HPF and DD HPF are run using 7 layers and 500 particles per layer, and the DD HPF$^+$ is run with a maximum of 8 layers (7+ refinement layer) and 500 particles per layer (recall that if no detections are found in a frame, the refinement layer is discarded). The 7 layers contain the variables related to torso (and head), left leg, right leg, left shoulder, right shoulder, left elbow and right elbow respectively. Since adding the refinement layer generally implies computing more particle weights, we also provide results for a DD HPF$^+$ using a total of 6 layers (5 + refinement layer) and 500 particles per layer. Using 5 layers implies filtering shoulder

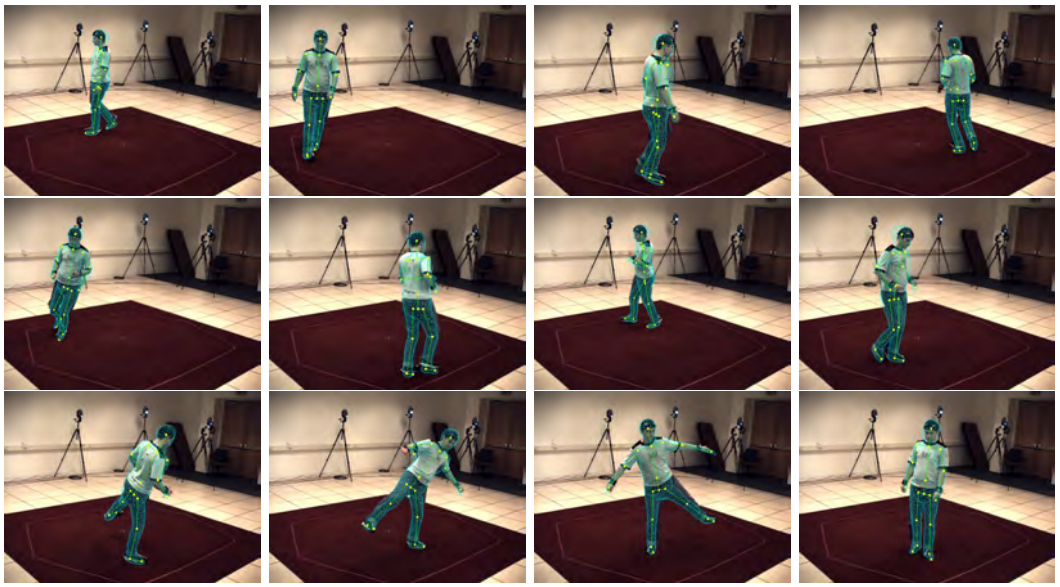


FIGURE 3.17: HPF + Local Optimization results for Subject S4 on the HumanEva II dataset.

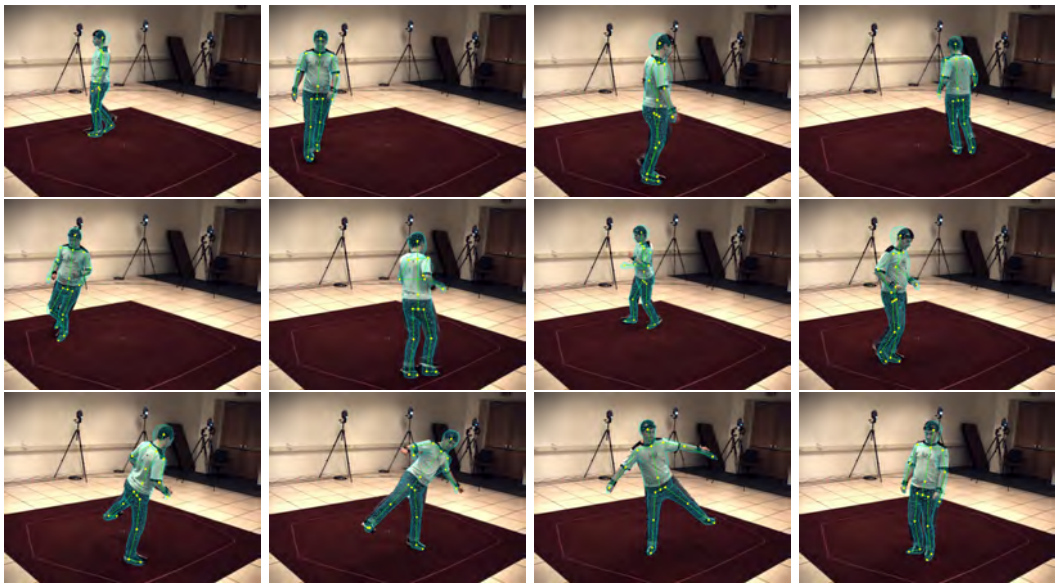


FIGURE 3.18: APO-HPF results for Subject S4 on the HumanEva II dataset.

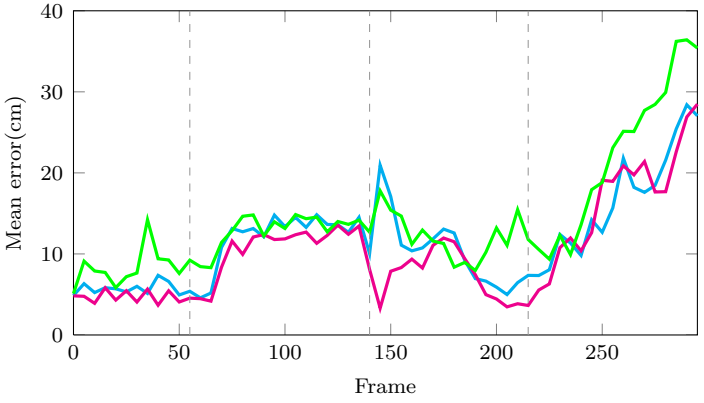
and elbow variables of one arm in the same layer.

Provided that in IXMAS dataset all the subjects have their hands uncovered, we employ the skin detector proposed by Jones and Rehg [JR99]. Despite of its simplicity, skin detection turns out to be a very effective approach towards tracking arm motion in this dataset. Recall that the detector driven framework can be applied to more sophisticated detectors.

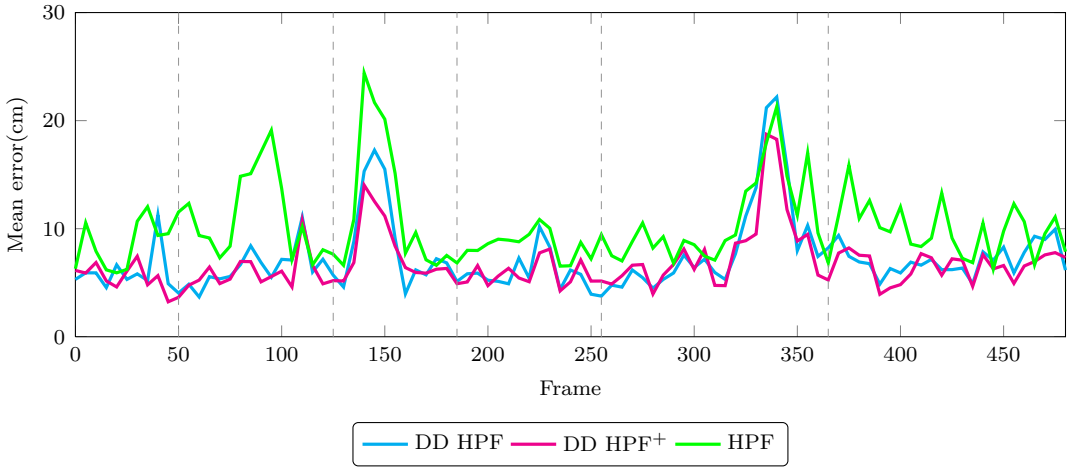
We compute the mean and standard deviation of the 3D error using all available ground truth data. Results are shown in Table 3.3.

Sequence (Frames)	APF	HPF	DD HPF	DD HPF$^+$	DD HPF$^+$ (5+1)
Alba 1 (53-350)	23.86 ±14.79	14.38±7.31	11.68±5.73	**10.32±6.02**	11.73±7.37
Alba 1 (658-1120)	17.13 ±7.93	11.84±6.87	10.86±6.03	**9.37±5.82**	10.38±7.96
Chiara 3 (29-292)	16.05 ±7.14	13.89±8.12	10.87±6.40	**9.85±4.43**	10.08±5.93
Chiara 3 (576-955)	19.16 ±10.16	11.29±5.81	8.36±5.88	**6.76±3.22**	7.98 ±5.89
Julien 1 (47-315)	18.30 ±8.37	15.18±7.21	13.86±6.85	13.30±6.85	**12.09±7.46**
Julien 1 (596-957)	26.01 ±14.70	11.85±6.06	9.43±3.63	7.87±2.73	**7.57±3.35**
Daniel 2 (15-306)	20.13 ±10.52	16.69±11.41	14.24±10.09	11.94±8.49	**11.74 ±7.16**
Daniel 2 (631-1119)	16.92 ±8.35	10.22±3.74	7.29±3.33	**6.73±2.53**	11.00 ±8.35
Srikumar 1 (43-368)	20.06 ±10.48	15.77±9.84	**14.20±11.22**	14.58±12.14	15.59 ±16.56
Srikumar 1 (704-1035)	18.59 ±10.23	13.60±9.99	13.46±8.47	**10.62±5.59**	12.07 ±8.87
Amel 1 (51-385)	20.30 ±9.32	16.00±8.02	16.76±8.47	**14.40±6.11**	15.08 ±5.88
Amel 1 (796-1295)	22.97 ±8.05	13.07±7.07	11.55±5.79	11.40±6.90	**11.35 ±5.99**

TABLE 3.3: Comparative results between the state-of-the-art methods and our proposals in the IXMAS dataset. We provide mean 3D error (and standard deviation) in centimeters. Bold figures highlight the result of the best method for each sequence.



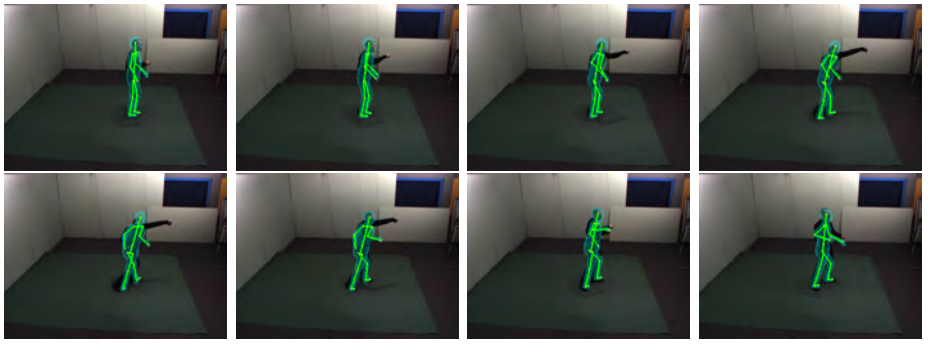
(a)



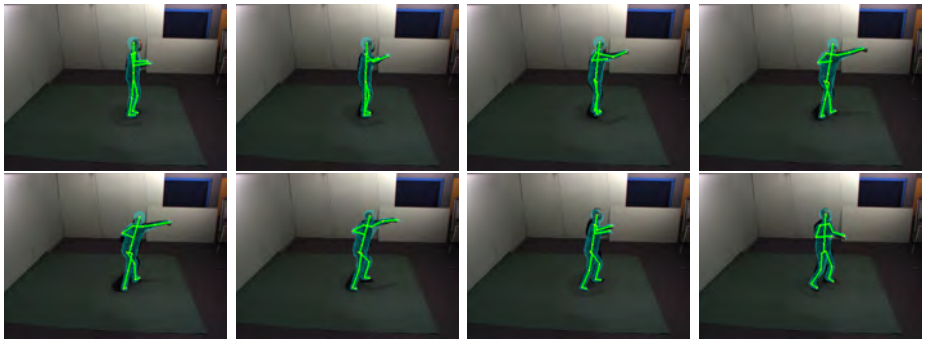
(b)

FIGURE 3.19: 3D pose error at every 5 frames for the HPF, DD HPF and DD HPF$^+$ for the IXMAS dataset (a) frames (53-350) of sequence alba1 (b) frames (631-1119) of sequence daniel2 . 7 layers and 500 particles have been used in the HPF and DD HPF whereas 8 layers and 500 particles per layer have been used in the DD HPF$^+$.

The proposed framework consistently outperforms both the APF and HPF. Apart from reporting a better accuracy in terms of mean error, the standard deviation is consistently reduced, thus reflecting the stability gain and the increased robustness in front of tracking failures. Experimental results show the effectiveness of the proposed DD HPF in reducing the blindness of the prior. In Figure 3.20, we provide a visual example for a fast arm action (action 6) and we compare the outcomes of the best state-of-the-art PF, the HPF, with the DD HPF . As we can see, the HPF gets lost whereas the DD HPF perfectly tracks the arm. The experimental validation also shows the impact of the refinement layer, which is able to filter erroneous particles originated by weaknesses of the silhouette-based observation model and erroneous classifications of extremities. Even when using less particle evaluations, the proposed method outperforms the state-of-the-art approaches.



HPF



DD HPF

FIGURE 3.20: Tracking example of subject 3 "punching"action (set 2) for the HPF and DD HPF using 7 layers and 500 particles per layer for both schemes.

3.8 Conclusions

In this chapter, we introduced a generic layered framework for human pose estimation. Human pose estimation problem is formulated as a non-linear non-Gaussian tracking problem. With such a formulation, we rely on particle filters in order to estimate the pose as the sample mean of a posterior distribution. In order to efficiently tackle the

curse of dimensionality, we advocate the use of a layered framework that generalizes existing PF algorithms. Under such a generalization we propose two major improvements. On the one hand, we propose a detector-driven propagation approach that incorporates 3D body part detections into the filtering formalism. Second, we develop an approximate partitioning of the input data that results in a faster and more accurate filtering approach. Our experiments in the publicly available Human Eva II and IXMAS datasets show the effectiveness of our approach.

4

Action Recognition with Nonlinear Dynamical Systems

4.1 Introduction

The recognition of human actions from video is a challenging problem with multiple applications in several fields, such as human-computer interaction, surveillance or gaming, to name a few. In the computer vision literature, the term action commonly denotes a simple motion pattern usually involving one subject [TCSU08]. Although such a term bounds the problem to some extent, human actions admit further taxonomies, depending on the kind of motion, on the stylistic performance and on the context, making the recognition problem elusive.

In recent years, computer vision researchers working on human action recognition have focused mainly on low-level features obtained from pixels [JSWP07, WS07, WKSL11]. These features usually encode information on appearance, hence the performance of systems relying on such features commonly suffers from changes in view-point, clutter or clothing. In earlier works, the problem of recognizing human actions was posed as one of modeling the dynamics of observed variables [CB95, PR00, BCMS01]. In such works, researchers usually relied on human pose and articulated motion estimates as an efficient way to encode motion. This kind of approaches are a minority compared to those working

with low-level features, since the problem of estimating human poses is also a challenging one with a traditionally higher cost. The previous chapter was devoted to bridge the gap between the low-level world (pixels) and the articulated model representation. In this chapter, we use the dynamics of articulated models of the human body in order to recognize actions, similarly to what earlier approaches pursued. Apart from the work in pose inference presented in this thesis, recent advances in human motion capture [SFC+11, GRBS10, BJB12] give extra motivation to the approach of this dissertation, i.e., to reconsider the analysis of human actions through pose estimates.

The approach followed in this chapter consists in modeling human actions by an underlying dynamical system. We exploit the theory of nonlinear dynamical systems and, more specifically, time-delay embeddings to model the dynamics of human actions using a set of parameters resulting from input data. Time-delay embeddings do not make assumptions on the form of the dynamical system, they basically employ input time-series to reconstruct a higher-dimensional Euclidean space that characterizes the underlying dynamical system topologically and geometrically. Consequently, we do not force any mathematical structure to the data, but we rather let the data speak by itself. We present a framework to exploit this principled modeling approach in computer vision. The idea is to build a link between the physics of dynamical systems with the semantics of action classes.

In this chapter, we focus on the recognition of actions using mocap data instead and pose estimates in order to analyze the potential of our approach. Furthermore, we show the feasibility of analyzing actions as time series obtained from low-level features.

4.2 Related Work

Some recent approaches on human action recognition have focused on modeling human actions as time-series and underlying dynamical systems. An early approach by Pavlovic and Rehg [PR00] infers the kinematics of walking and running to model dynamical systems that can be used to recognize both actions. With similar kinematic features, Bisacco et al. [BCMS01] classify human gaits by defining a distance in the space of dynamical systems. Campbell and Bobick [CB95] represent human motion using joint angles to construct curves in a phase space. Their approach is able to segment fundamental steps in ballet dance. Lv et al. [LNL05] recognize human actions by representing them as a set of weighted channels consisting in the temporal evolution of 3D joint coordinates. Raptis et al. [RWS10] propose a method to model human actions as a linear time-invariant dynamical model of relatively small order that generates multivariate time series in the form of joint angle dynamics along time.

Some authors focus on dimensionality reduction methods to deal with the action recognition problem by embedding input feature spaces into low-dimensional spaces. Dimensionality reduction methods have been widely applied with low-level and appearance features [CWSS07][WS07][BR07], due to the high dimensionality of these features. However, when one deals with actions and time series, keeping the geometric structure of input spaces is often not sufficient, and there is a need for constraining embedding techniques to cope with the underlying dynamics. Lewandowski et al. [LMdRMN10], propose a Temporal Laplacian Eigenmaps to embed time series of motion capture data and video data, obtaining good results on both human motion reconstruction and action recognition. Similarly, Wang et al. [WFH08] constrain manifold learning to be dynamically consistent with the input space.

Some recent approaches aim at overcoming modeling limitations produced by assumptions made on the dynamical model. Li et. al [LAM+11] model actions as the output of an unknown dynamical system using the distance between principal angles of subspaces representing different action classes. With the same objective, Ali et al. [ABS07] propose to exploit the theory of chaotic systems to recognize human actions. They embed one-dimensional time-series, each of which being the evolution of a joint coordinate, in an m-dimensional phase space. Using metric and dynamical invariants of the reconstructed phase space, they recognize actions using MoCap data and video data. In a recent work, Basharat and Shah [BS09] extend the embedding of univariate signals to multivariate analysis in order to synthesize human motion and dynamic textures. The approaches presented in this thesis are based on this latter sort of nonlinear dynamical models [KS04], and more specifically on time-delay embeddings.

4.3 Body Model as a View-Invariant Action Representation

In the proposed approach, human actions are seen from a dynamical systems perspective, hence requiring a time-varying signal as input data. This actually implies that many different features could be employed. In this paper, we focus on using an articulated body model for human action recognition, as one can then represent human actions in a view-invariant manner [PC06]. With this modeling framework, and provided that bone elongations are fixed, human pose is represented by a set of kinematic features: a translation $\mathbf{r} \in \mathbb{R}^3$, a global rotation $\phi \in \mathbb{R}^3$ and a set of joint angle rotations $\theta_i \in \mathbb{R}$. Each joint can have up to 3 associated rotations, depending on its degrees of freedom, and these rotations are independent of the global translation and rotation. Using this

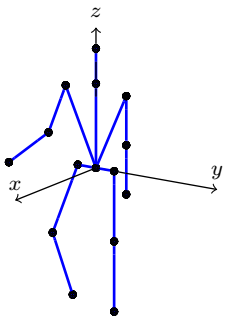


FIGURE 4.1: Example of articulated human body model and its local coordinate axes

representation, one can alternatively represent the pose as a set of joint positions $\mathbf{p}_j \in \mathbb{R}^3$.

Due to the independence of joint rotations with respect to global variables, we can represent human motion in a view invariant manner by discarding those model variables whose evolution along time does not characterize the type of action. For instance, when humans walk they can go in several directions. These directions are represented by the global orientation (rotation along z axis in Fig. 4.1) and the translation in the transverse plane (xy in Fig. 4.1). By discarding these variables, which are global variables, one can effectively represent walking motion as if humans where in a treadmill, thus achieving invariance to the directions of walking, which, equivalently, means view-invariance. However, not every global variable will be meaningless, in the sense of view-dependence, for action recognition. And, similarly, not all the available rotations and positions will provide characteristic information of an action. For that matter, methods involving feature selection or feature weighting will be necessary in order to discard irrelevant variables.

For the rest of the chapter, we consider that we have a motion sequence represented by an articulated skeleton whose pose changes along time. Consequently, at every time instant t, we will be able to compute the composite feature vector $\{\mathbf{r}, \phi, \theta_1, \ldots, \theta_K, \mathbf{p}_1, \ldots, \mathbf{p}_P\}$. This composite vector is a high-dimensional feature vector and its components can be seen as 1-dimensional time series. Throughout the chapter, we will label each of these 1-dimensional components with superindex ξ. Formally, this view of the articulated model variables at time t is noted as:

$$\{z^1(t), \ldots z^\xi(t), \ldots, z^\Xi(t)\} \tag{4.1}$$

and the time series of one variable are noted as:

$$\mathbf{z}^\xi = \{z^\xi(1), \ldots z^\xi(t), \ldots, z^\xi(T)\} \tag{4.2}$$

4.4 Action Recognition in the Phase-Space

In this section we present a framework where human actions are modeled by an underlying dynamical system. We exploit the theory of nonlinear dynamical systems and, more specifically, time-delay embeddings to model the dynamics of human actions using a set of parameters resulting from input data. Compared to other approaches [RWS10], time-delay embeddings do not make assumptions on the form of the dynamical system, they basically employ input time-series to reconstruct a higher-dimensional Euclidean space that characterizes the underlying dynamical system topologically and geometrically. We then assume that a human action class is a realization of a set of reconstructed dynamical systems, and that such realization will visit the same regions of the reconstructed spaces. Under this assumption, we propose several methods to measure the geometric and topological similarities in the reconstructed spaces such that actions can be effectively recognized.

In the following, we describe the proposed approach. We start by detailing the time-delay embedding methodology employed in this thesis in order to reconstruct the phase-spaces of each 1-dimensional time series. Then, we present several classification approaches that exploit the proposed embedding.

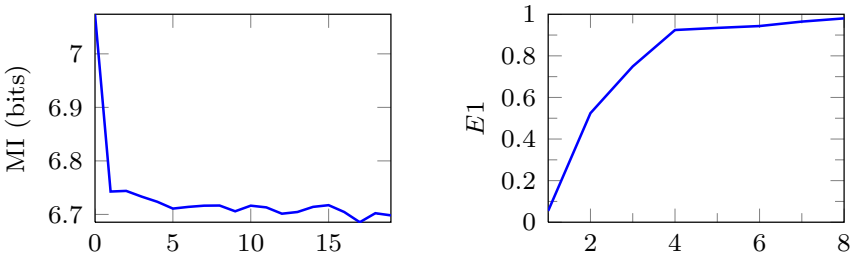
4.4.1 Time-Delay Embedding

Let us define the underlying dynamical system of a human part or model variable motion ξ as the possibly nonlinear map $f : \mathbb{R}^m \to \mathbb{R}^m$ that describes the temporal evolution of the state variables $\mathbf{x}^\xi(t) = [x_1^\xi(t), x_2^\xi(t), \ldots, x_m^\xi(t)] \in \mathbb{R}^m$. As the state space variables evolve, they generate one-dimensional observables $z^\xi(t)$ that constitute the action time series of interest. These time series are a temporally ordered collection of part locations or variables evolving in time. Using $z^\xi(t)$, we want to characterize the underlying system f by reconstructing an m-dimensional space where time series are unfolded.

To reconstruct this space, we rely on the theorem by Takens [Tak81], that provides theoretical justification for reconstructing state spaces using simple time-delay embeddings:

$$\hat{\mathbf{x}}^\xi(t) = [z^\xi(t), z^\xi(t+\tau), \ldots, z^\xi(t+(m-1)\tau)] \tag{4.3}$$

where $\hat{\mathbf{x}}^\xi(t)$ is a delay-vector (a point in the reconstructed space), m is the embedding dimension and τ is the embedding delay. This reconstructed space, also called phase space, is a metric space. According to [Tak81], for a sufficiently large m, this space is an homeomorphism of the true dynamical system that generated the time series. Hence,



(a) Embedding Delay (b) Embedding Dimension

FIGURE 4.2: Determining the embedding delay and a suitable embedding dimension for a joint angle scalar time series. (a) Mutual information of the series and its delayed version. The first minimum is used as the optimal embedding delay. (b) $E1$ measure, obtained by Cao's method [Cao97]. Suitable embedding dimensions are found when the $E1$ measure starts converging to a stable, high value. In the example, $m = 4$ is a sufficient embedding dimension.

by stacking sets of m temporally equispaced samples of the input scalar time series we are able to map the input series onto meaningful sets of delay-vectors.

While dimension is important from a theoretical viewpoint, the embedding delay τ has shown to be of relevance from a practical viewpoint: low delays might reconstruct spaces with improperly unfolded time series due to high correlated samples, while high delays might cause a loss of information about the initial state. We first determine the embedding delay using the mutual information method [FS86] and then we employ the estimated delay to find the appropriate embedding dimension using the method by Cao [Cao97]. We detail both methods for completeness.

Embedding delay The idea behind the method for inferring the embedding delay is that an appropriate embedding delay must minimize the dependency between the delay-vector coordinates, so that the time series data gets effectively unfolded in a higher dimensional space. Although one could employ zeros of the autocorrelation function in order to analyze the dependency between samples $z(t)$ and $z(t+\tau)$, the method in [FS86] shows that relying on the mutual information is a better criterion. The optimal delay is obtained by taking the first minimum of the mutual information (MI) function between the original time series and a delayed version of the same time series (see Algorithm 7 and Fig. 4.2a). This MI-based method for finding a suitable embedding delay has also the interesting property of providing, to some extent, temporal invariance. In other words, if the same action is performed at a different speed, the first minimum of the MI will be displaced accordingly.

Embedding dimension Several methods for computing a suitable embedding dimension with experimental data rely on the false nearest neighbors idea formulated by

Algorithm 7: Embedding Delay

$\mathbf{z}^{\xi} = \{z^{\xi}(0), ..., z^{\xi}(T-1)\}$: Time series obtained from the variable ξ
Normalize $\mathbf{z}^{\xi}$ between 0 and 1
Set $minMI = C \gg 1$
for $\tau = 1 \dots max\ \tau$ **do**
  $\mathbf{z}^{\xi}_{\tau} = \{z^{\xi}(\tau), ..., z^{\xi}((T-1+\tau) \mod (T))\}$: Delayed time series
  $I(\tau) = -\sum_{\mathbf{z}^{\xi}} \sum_{\mathbf{z}^{\xi}_{\tau}} p(\mathbf{z}^{\xi}, \mathbf{z}^{\xi}_{\tau}) \log \frac{p(\mathbf{z}^{\xi}, \mathbf{z}^{\xi}_{\tau})}{p(\mathbf{z}^{\xi}) p(\mathbf{z}^{\xi}_{\tau})}$
  if $I(\tau) < minMI$
  then
    $minMI = I(\tau)$
  end
  else
    break
  end
end
$\tau_{opt} = \tau$

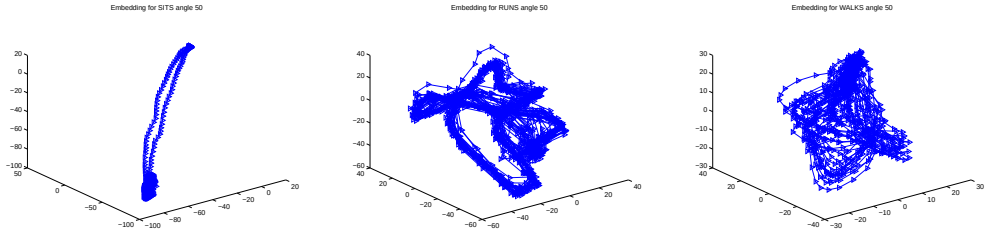


FIGURE 4.3: Examples of reconstructed 3-D spaces. Hip rotations for sit, run and walk action instances of the FutureLight dataset [Fut].

Kennel et al. [KBA92]. They observed that the phase space of a dynamical system folds and unfolds smoothly as dimensions vary. Hence, two points that are sufficiently close in a given m-dimensional reconstructed phase space, should keep close in $m+1$ dimensions. As a sufficient embedding dimension is reached, the number of false nearest neighbors (points that are nearest neighbors in m dimensions but not in $m+1$ dimensions) should be negligible. Based on this seminal idea, Cao [Cao97] proposed an improved method for finding a sufficient embedding dimension (see Algorithm 8 and Fig. 4.2b) that overcomes the limitations of the original method by Kennel et al. [KBA92] when dealing with short-term experimental data.

Once both the embedding delay and the embedding dimension have been estimated, the reconstructed space takes the following form:

$$\hat{\mathbf{X}}^{\xi} = \begin{bmatrix} z^{\xi}(0) & z^{\xi}(\tau) & \cdots & z^{\xi}((m-1)\tau) \\ & \vdots & & \\ z^{\xi}(t) & z^{\xi}(t+\tau) & \cdots & z^{\xi}(t+(m-1)\tau) \\ & \vdots & & \end{bmatrix} \tag{4.4}$$

Algorithm 8: Embedding Dimension

$\hat{\mathbf{x}}^{\xi}_{m}(i) = \left[z^{\xi}(i) \quad z^{\xi}(i+\tau) \quad \cdots \quad z^{\xi}(i+(m-1)\tau)\right]$: Reconstructed delay-vector in m dimensions
N: Number of points in time series $\mathbf{z}^{\xi}$
for $m = 1 \ldots M$ **do**
  for $i = 0 \cdots N-1-m\tau$ **do**
    Find the Nearest Neighbor $\hat{\mathbf{x}}^{\xi}(j)$
    $a(i,m) = \frac{\|\hat{\mathbf{x}}^{\xi}_{m+1}(i) - \hat{\mathbf{x}}^{\xi}_{m+1}(j)\|}{\|\hat{\mathbf{x}}^{\xi}_{m}(i) - \hat{\mathbf{x}}^{\xi}_{m}(j)\|}$
  end
  $E(m) = \frac{1}{N-m\tau} \sum_{i=0}^{N-1-m\tau} a(i,m)$
  if $m \geq 2$ **then**
    $E1(m-1) = E(m)/E(m-1)$
  end
  if $m \geq 3$ **then**
    if $\|E1(m-1) - E1(m-2)\| < th$ **then**
      break
    end
  end
end
$m_{opt} = m-1$

where each row of the matrix $\hat{\mathbf{X}}^{\xi}$ is a delay-vector. The reconstructed space $\hat{\mathbf{X}}^{\xi}$ (see Fig. 4.3) can be used to compute chaotic invariants [ABS07] for human action recognition. In this thesis, however, we present a different analysis of the reconstructions and how this analysis helps in recognizing human actions.

Note that time-delay embeddings are performed separately for each 1-dimensional time series. The reason for embedding univariate time series is motivated by several reasons. First, human actions may be characterized by a few relevant motion patterns taking place in specific parts. For instance, motion and pose patterns taking place in the arm are expected to be more relevant for the class *raise hand* than for the class *kick*. Under this assumption, we expect a classifier to have more chances of removing irrelevant motion patterns if we separately embed univariate time series than if we couple all together in the input space before computing the embedding. This intuitive reasoning matches the empirical findings by Raptis et al. [RWS08]. They report that the performance of an action recognition method based on time series analysis is better when the action models are built from univariate time series. Second, coupling time-series in the input space may yield an dramatical increase in the dimensionality of the feature space, as many combinations might be possible and useful. And third, an extension proposed by Cao et al. [CMJ98] considers the possibility of coupling embedded uni-dimensional time-series in a joint space. Yet simple, this extension has proved to be effective for the prediction of multivariate time series [BS09].

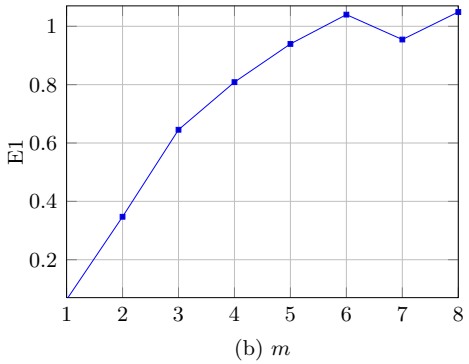


FIGURE 4.4: Minimum Common Embedding dimension. Example of averaged E1 measure [Cao97]. In this example, $m = 5$ can be considered the minimum common embedding dimension, since E1 has almost settled down to its upper bound.

4.4.2 Minimum Common Embedding Dimension

Differently to a few existing methods of the state-of-the-art [ABS07, BS09], we propose to view the embedded spaces as a unique Euclidean space where all the time series are unfolded. Consequently, we need to find the dimensionality of such space. According to Takens' theorem [Tak81], the dynamical system can be embedded in an Euclidean space with dimension larger than twice of the system attractor. This means that there is not a unique embedding dimension, but a minimum sufficient embedding dimension, that can be found by empirical methods such as the already described Cao method [Cao97]. For that matter, a minimum common embedding dimension between a set of time series $\mathbf{z}^{\xi}$ must exist. This minimum common embedding dimension is the minimum dimension for which the whole set of time series gets effectively unfolded.

The minimum common embedding dimension will be employed in order to embed new unseen motion sequences, and hence can be found by means of cross-validation on training data. Alternatively, we propose to use a false nearest neighbor method on all the available training data in order to find the minimum common embedding dimension. In order to estimate the minimum common embedding dimension from training data, we use Cao's method in all the available motion sequence. In order to correctly unfold the data from a single dynamical system, nearest neighbors are computed within each motion sequence. In this way, we avoid mixing embedded points coming from motion sequences that might have been generated by a different dynamical system. By averaging the E1 measure for each embedding dimension we estimate the unfolding rate of the delay-vectors (see Fig. 4.4). The minimum common embedding dimension is the value of m such that the E1 measure settles down, getting close enough to an upper bound (usually close to 1). In practice, we select M by thresholding the rate of change of the E1 measure when moving from m to $m+1$ (see Algorithm 8). Whenever $E1(m+1) > 0.9$, we check if $E1(m+1) - E(m)$ is below 0.1.

By using a common embedding space, we facilitate the use of Euclidean metrics on the reconstructed spaces, which at the same time pave the way to the development of recognition methods based on these metrics. Another advantage is that we do not need to estimate the embedding dimension in test time. Furthermore, a common embedding dimension opens the possibility to use features relating two model variables in the embedded space, similarly to what relational features [MRC05] do in the input space.

4.4.3 Recognition by Trajectory Matching in Phase-Spaces

The first proposed recognition method is based on comparing the set of reconstructed phase portraits of a target action (each one obtained from univariate time series) with a set of phase portraits from representative templates. In order to perform such a comparison, we propose a distance between two reconstructed phase portraits $\hat{\mathbf{X}}_a^\xi$ and $\hat{\mathbf{X}}_b^\xi$.

We use the map $f(\mathbf{x}(t)) \rightarrow \mathbf{x}(t{+}1)$ to refer to the underlying dynamical system associated to some model variable and $\hat{f}(\mathbf{x}_a(n)) \rightarrow \mathbf{x}_a(n+1) \quad \mathbf{x}_a \in \hat{\mathbf{X}}_a^\xi$ to the system representing the evolution in the phase portrait $\hat{\mathbf{X}}_a^\xi$ reconstructed with the model time series (we use n instead of t to denote that we have a sampled signal). From this notation we introduce the nonlinear time evolution operator $\hat{f}^n$ such that:

$$\hat{f}^0 = \text{identity} \tag{4.5}$$

$$\hat{f}^1 = \hat{f} \tag{4.6}$$

$$\hat{f}^n \hat{f}^s = \hat{f}^{n+s} \tag{4.7}$$

Due to the nature of human motion, we assume that f is smooth. We also assume that important topological characteristics of the underlying dynamics are reflected on the trajectories of the reconstructed phase portraits (Fig. 4.3), in spite of the fact that the data employed is a sampled version of a continuous signal.

Our proposal for the distance between reconstructed phase portraits relies on these properties and it is inspired on DTW [BC96]. The underlying idea is to traverse one phase portrait $\hat{\mathbf{X}}_a^\xi$ and look for the nearest neighbor at each time instant in another phase portrait $\hat{\mathbf{X}}_b^\xi$. If the portraits have been generated by similar motion (and possibly by the same action) nearest neighbors should be in closer distances along the respective reconstructed phase portraits. Nearest neighbor retrieval is, therefore, restricted in such a way that the temporal ordering is preserved. This is equivalent to measuring

the prediction error of points in $\hat{\mathbf{X}}_a^\xi$ given $\hat{\mathbf{X}}_b^\xi$. Algorithm 9 describes the proposed computation of the distance between two reconstructed phase portraits of some body model variable. The proposed distance is posed as a prediction error, but can be seen as an efficient alignment between two phase-space trajectories of (possibly) different length. We experimentally compared our algorithm with a more exhaustive search for phase-space trajectory alignment, and we found that the proposed distance provides better classification results.

Algorithm 9: d($\hat{\mathbf{X}}_a^\xi$, $\hat{\mathbf{X}}_b^\xi$)

Start at $\mathbf{x}_a(0) \in \hat{\mathbf{X}}_a^\xi$
$\mathbf{x}_{bl} = \mathbf{x}_b(0)$
$d = \|\mathbf{x}_a(0) - \mathbf{x}_b(0)\|^2$
$n = 1$
while $\mathbf{x}_a(n) \neq \mathbf{x}_a(0)$ **do**
  $l = \underset{k}{argmin} \quad \|\mathbf{x}_a(n) - \hat{f}_b^k(\mathbf{x}_{bl}))\|^2$ s. t. $0 \leq k < \delta$.
  $d = d + \|\mathbf{x}_a(n) - \hat{f}_b^l(\mathbf{x}_{bl}))\|^2$
  $\mathbf{x}_{bl} = \hat{f}_b^l(\mathbf{x}_{bl})$
  $n = n + 1$
end
$d = d/n$

The search for nearest neighbors is restricted by δ to the near future of the phase portrait. This variable aims at compensating for the different lengths of the available action samples. Note that while the distance performs a one-pass traversal through $\hat{\mathbf{X}}_a^\xi$, we allow $\hat{f}_b$ to go from the last reconstructed state space point to the first one. This is clearly necessary for periodic or quasi-periodic human motion, such as walking. Note also that the distance between $\hat{\mathbf{X}}_a^\xi$ and $\hat{\mathbf{X}}_b^\xi$ is, in general, different from the distance between $\hat{\mathbf{X}}_b^\xi$ and $\hat{\mathbf{X}}_a^\xi$. Hence, in order to have a true metric, we define the final distance between two phase portraits as:

$$e(\hat{\mathbf{X}}_a^\xi, \hat{\mathbf{X}}_b^\xi) = \frac{d(\hat{\mathbf{X}}_a^\xi, \hat{\mathbf{X}}_b^\xi) + d(\hat{\mathbf{X}}_b^\xi, \hat{\mathbf{X}}_a^\xi)}{2} \tag{4.8}$$

4.4.4 Training and Recognition

Suppose we are given a labeled training data set $\mathcal{Y} = \{(\mathbf{Z}_1, y_1), \cdots, (\mathbf{Z}_Q, y_Q)\}$ consisting in multivariate time series $\mathbf{Z}_i = \{\mathbf{z}_i^1, \cdots, \mathbf{z}_i^\xi, \cdots, \mathbf{z}_i^\Xi\}$ representing an action instance with label y_i. We propose two approaches that employ the trajectory matching framework to classify actions from these time series. The first one employs Support Vector Machines (SVMs) [Vap95] and averages the distances between each model variable ξ. The second one assumes that only a subset of model variables are relevant and consists in learning

the importance of each variable to efficiently fuse their trajectory similarities in test time.

The first step, common to both methods, is to find minimum common embedding dimension m in order to embed the training data $\mathcal{Y}$ into a common phase-space $\mathcal{X} = \{\hat{\mathbf{X}}_1^1, \ldots, \hat{\mathbf{X}}_1^\xi, \ldots, \hat{\mathbf{X}}_Q^\xi, \ldots, \hat{\hat{\mathbf{X}}}_Q^\Xi\} \in \mathbb{R}^m$.

SVM Approach: We define a feature vector gathering the degree of similarity between all the available training samples. To this end, we define the similarity in terms of the distance between the reconstructed phase spaces for each variable ξ. That is, we compute the proposed distance (see Section 4.4.3) between the reconstructed phase space of each available pair of examples , namely $\hat{\mathbf{X}}_i^\xi$ and $\hat{\mathbf{X}}_j^\xi$. The similarity between i and j is then given by the average of all the similarities between joint angle trajectories:

$$s_{ij} = \frac{1}{N_\xi} \sum_\xi \exp\left(-e(\hat{\mathbf{X}}_i^\xi, \hat{\mathbf{X}}_j^\xi)\right) \tag{4.9}$$

In this way, for each training sequence i we have an associated feature vector gathering all the similarities with respect to the Q available training sequences, $\{s_{i1}, \ldots, s_{iQ}\}$, and each test sequence ρ has an associated feature vector $\{s_{\rho 1}, \ldots, s_{\rho Q}\}$. We train a RBF-SVM with the Q training vectors, in order to perform the subsequent classification of test vectors.

Weighted Approach: Human actions are not necessarily characterized by full body motion. Clearly, understanding the action *raise hand* does not require knowledge of the motion of legs. Even in the case of having full body motion, it might happen that some parts are more important for understanding a human action. Motivated by the potential saliency of different body parts in different actions, we use training data to learn a set of weights reflecting the importance of the motion of different body model variables in each action class.

Due to the independence of the variables in the model-based representation of human actions, we propose a strategy that performs fusion of the similarity scores obtained in each model variable ξ. We aim at maximizing the classification accuracy by weighting each one of the variables ξ based on their discriminative power on the training data. For this reason, we first classify each training sample using the nearest neighbor, based on the distance $e(\hat{\mathbf{X}}_i^\xi, \hat{\mathbf{X}}_j^\xi)$, $i \neq j$. Using the training data false positive, false negative (misses) and true positive classifications for the action class label y in each model variable

ξ ($\mathrm{fp}_{Y_\xi=y}$, $\mathrm{fn}_{Y_\xi=y}$ and $\mathrm{tp}_{Y_\xi=y}$ respectively), we define the weights for each variable ξ and action y as:

$$w_{Y_\xi=y} \propto \frac{\mathrm{tp}_{Y_\xi=y}}{\mathrm{fp}_{Y_\xi=y} + \mathrm{fn}_{Y_\xi=y} + \mathrm{tp}_{Y_\xi=y}} \tag{4.10}$$

To classify a test action based on the set of reconstructed phase portraits $\hat{\mathbf{X}}_\rho^\xi$, we normalize the weights $w_{Y_\xi=y}$ such that $\sum w_{Y_\xi=y} = 1$, and we fuse the similarity scores with respect to the training samples :

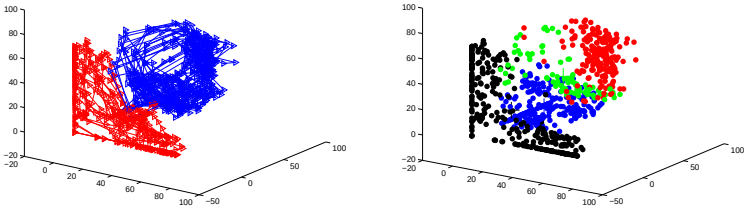
$$\hat{y} \quad = \quad \underset{y \in Y}{argmax} \quad \sum_{\xi} w_{Y_\xi=y} \exp\left(-e(\hat{\mathbf{X}}_\rho^\xi, \hat{\mathbf{X}}_{i,Y_\xi=y}^\xi)\right) \tag{4.11}$$

The proposed strategy takes advantage of training data to learn the variable model weights in a discriminative manner, i.e., the importance of each model variable depends on the extent to which its motion pattern in a given action differs from motion patterns in other actions.

4.4.5 Recognition by Bag of Delay-Vectors model

When matching trajectories, a basic problem arises: if a huge number of templates is required, one has to compute the distance between phase portraits for each training sample in the template dataset. Hence, finding an alternative way of exploiting the embedded space, without requiring exhaustive distance computation is of interest for many test scenarios.

To overcome exhaustive distance computations, we formulate two basic assumptions. First, human actions are driven by dynamical systems that exhibit some transient behavior, hence state space points will eventually lie on an attracting set. Second, since time-delay embeddings reconstruct spaces that are homeomorphisms of an underlying dynamical system, these attracting sets will be approximately represented by reconstructions with experimental data. These assumptions are also taken into account for the trajectory matching in phase spaces. However, here we relax the conditions and we do not formulate any requirement on the topology. As before, in order to jointly characterize and compare the attracting sets for all the actions, we propose to find a common embedding space for the time series. After mapping the input data onto a common higher-dimensional space, we use histograms of reconstructed delay-vectors to geometrically characterize attracting sets. This histogram-based approach, termed Bag



(a) (b)

FIGURE 4.5: Bag of Delay-Vectors concept (best viewed in color). (a) 3-D embedding for knee rotation time-series obtained with run(blue) and walk(red) action classes and (b) clustered delay-vectors. Attracting sets for each motion are represented by different sets of clusters.

of Delay-Vectors (BoDV), requires clustering delay-vectors and choosing a representative point, that will act as a *word* (see Fig. 4.5).

Suppose we are given a labeled training data set $\mathcal{Y} = \{(\mathbf{Z}_1, y_1), \cdots, (\mathbf{Z}_Q, y_Q)\}$ consisting in multivariate time series $\mathbf{Z}_i = \{\mathbf{z}_i^1, \cdots, \mathbf{z}_i^\xi, \cdots, \mathbf{z}_i^\Xi\}$ representing an action instance with label y_i. We first find the minimum common embedding dimension for the training data. Note that this only applies for the embedding dimension, but not for the embedding delay, which is computed for each univariate time series following the MI method [FS86]. After applying the appropriate time-delay embedding for all the available time series of a part or variable $\xi = j$, delay-vectors belonging to a specific model variable or part motion $\mathbf{X}^{\xi=j}$ can be represented in the same space $\mathcal{X}^{\xi=j} = \{\hat{\mathbf{X}}_1^{\xi=j}, \cdots, \hat{\hat{\mathbf{X}}}_Q^{\xi=j}\} \in \mathbb{R}^m$. We cluster the delay-vectors in each part-specific space $\hat{\mathcal{X}}^{\xi=j}$ and we compute histograms associated to each $\mathbf{X}_i^{\xi=j}$ obtained from input time series $\mathbf{z}^{\xi=j}$. Finally, a feature vector for an instance $(\mathbf{Z}_i, y_i)$ is obtained by stacking the Ξ part-specific histograms belonging to the same action instance.

Provided that the feature vectors are high-dimensional BoDV models, we employ a supervised SVM approach [Vap95] with χ^2-kernel. We train multi-class SVM models with the stacked BoDV built on the training data set. To classify human actions, we embed the test action sequences, we quantize the delay-vectors and we construct an appropriate feature vector by stacking histograms computed with the quantized delay-vectors.

We left out the details of the clustering step, which is an important element of our approach. For the sake of simplicity, we employ k-means clustering, which a priori suffices to obtain a geometric characterization of the reconstructed space. Nonetheless, k-means is not an optimal clustering method for delay-vectors, whose attracting sets might not be properly explained by spherical clusters. Moreover, if time series are not properly unfolded by time-delay embeddings, one might seriously damage the statistical relation

between the clusters and the input time series [KL03]. Alternatively, we use a hierarchical agglomerative clustering approach [War63]. Since clusters are not represented by its mean, but by the point population, we select k clusters and we model the delay-vector distribution in each cluster by a Gaussian distribution. In this way, we expect to obtain a more accurate geometric characterization of attracting sets. Quantization of test vectors is performed by finding the most probable cluster and then using the mean delay-vector as a word.

4.5 Forests in the Phase Space

The use of codebooks of delay-vectors is a more efficient way of handling time series data from multiple model variables. Nonetheless, the clustering step becomes crucial in order to geometrically characterize the phase-space. As discussed in the previous section, clustering methods such as k-means clustering may not provide accurate results for delay-vectors. Furthermore, clustering methods employed so far are fully unsupervised, hence the result might not yield a discriminative enough codebook for action recognition. Another issue, shared by trajectory matching and BoDV, is the fact that we do not exploit the common embedded space as the phase-space of the multivariate time series comprising the evolution of the variables in the whole articulated model [BS09].

We address these issues by proposing a novel action recognition framework that performs a supervised partition of the joint phase-space. Provided that the joint phase-space is a highly dimensional Euclidean space, we use random forests [Bre01] in order to find the partitions that, given some training data, provide the best separation of action classes.

Our approach relies on multivariate phase space reconstruction. So far, we have used time-delay embeddings with univariate time series, where each series captures the temporal evolution of a model variable ξ. In order to embed multivariate time series, Cao et al. [CMJ98] proposed a practical extension consisting in stacking together the embedded phase spaces of separate univariate time series:

$$\hat{\mathbf{X}}(t) = \begin{bmatrix} z^1(t) \cdots z^\xi(t+\tau_1) \cdots z^\xi(t+(m-1)\tau_1) \\ \vdots \\ z^\xi(t) \cdots z^\xi(t+\tau_\xi) \cdots z^\xi(t+(m-1)\tau_\xi) \\ \vdots \\ z^\Xi(t) \cdots z^\Xi(t+\tau_\Xi) \cdots z^\Xi(t+(m-1)\tau_\Xi) \end{bmatrix} \tag{4.12}$$

Contrarily to other approaches [CMJ98, BS09], we employ this extension making use of the minimum common embedding dimension m proposed in this dissertation. This choice allows us to use relational features [MRC05] in the embedded space.

With the proposed embedding, the training data in the embedded space is noted as $\mathcal{Y} = \{(\hat{\mathbf{X}}_1, y_1), \cdots, (\hat{\mathbf{X}}_Q, y_Q)\}$ where $\hat{\mathbf{X}}_i$ are the embedded multivariate time series of the i-th human motion sequence and y_i is the action label corresponding to the sequence.

For the sake of completeness, in the following we provide a detailed description of random forests. Random forests [Bre01] are an ensemble of m decision trees, which are binary trees. Nodes n in each tree have a learned probability distribution $p_n(y|\hat{\mathbf{X}}, t)$ that reflects how likely is a class y given a time instant t and the features $\hat{\mathbf{X}}$ (in this case, delay-vectors in the common phase space reconstructed from multivariate time series). These probability distributions are learned by recursively branching left or right down the tree, according to some node-specific weak classifier, until some stopping criteria are met and thus a leaf node is reached. Weak classifiers associated to each node are binary functions of feature vectors obtained from the reconstructed phase space $\hat{\mathbf{X}}$. The robustness of forests is based on the combination of several decision trees. Usually, one performs this combination by averaging the distributions over the leaf nodes $\{l_1, \cdots, l_M\}$ reached in all the M trees:

$$p_(y|\hat{\mathbf{X}}, t) = \frac{1}{M} \sum_{m=1}^{M} p_{l_m}(y|\hat{\mathbf{X}}, t) \tag{4.13}$$

Each tree is trained separately with a small subset of the training data obtained by sampling with replacement. Learning is based on the recursive splitting of training data into left $\mathcal{L}$ and right $\mathcal{R}$ subsets, according to some binary test f and a threshold θ. The binary test is a function of the feature vectors obtained from each training example. In the particular case of the classification of temporal events, training examples are obtained from patches of a given duration centered at given time instants.

At each node, a test f and a threshold θ are randomly generated, and the one that maximizes some criteria is selected. We employ the information gain as a test selection criterion:

$$\Delta IG = -\frac{|\mathcal{L}|}{|\mathcal{L}| + |\mathcal{R}|} H(\mathcal{L}) - \frac{|\mathcal{R}|}{|\mathcal{L}| + |\mathcal{R}|} H(\mathcal{R}) \tag{4.14}$$

where $|\ |$ denotes the number of elements of the subset and $H()$ is the Shannon entropy of the classes in a subset. The process continues until a maximum depth D is reached or the information gain cannot be further maximized.

The outcome of this training process is an ensemble of decision trees, each of which defines a partition of the feature space that aims at maximizing the separation of classes. In test time, an input patch is tested down each decision tree until it achieves the respective leafs. Class probabilities from the leafs are averaged, obtaining the probability distribution $p_(y|\hat{\mathbf{X}}, t)$. This probability distribution constitutes a vote, where the vote strength is given by the probability of each individual class y. The averaging of the class probabilities in each one of the reached leafs assures better generalization thanks to the maximum-margin classification achieved by random forests [CSK12].

We propose a family of binary tests in order to classify actions in the phase space. These binary tests aim at computing relational figures between delay-vectors of a given patch centered at time t. In fact, and thanks to the multivariate phase-space reconstruction, our proposed binary tests can deal with the input space and admit combinations of Euclidean spaces of different dimensionality.

According to the representation of the phase-space in Eq. 4.12, the reconstructed phase-space, using the multivariate extension, can be seen as a tensor whose first mode is the temporal dimension, the second mode is the variable ξ (rows in Eq. 4.12) and the third mode is the coordinate of the delay-vector (columns in Eq. 4.12). If we let χ denote the third dimension of the tensor, we then can access any variable by combinations of (ξ, χ). The most interesting case, however, is that allowing us to recover delay-vectors of the phase space and scalars or vectors from the input space. For instance, we can deal with the set of joint angles in $\mathbb{R}$ and its corresponding phase space of common embedding dimension m (note that here the dimension of the Euclidean space refers to the space where points lie, rather than to the length of the feature vector). In such a case, $\hat{\mathbf{X}}(t, :, 1)$ is the column vector gathering the input space, i.e, joint angles at time t. Similarly, $\hat{\mathbf{X}}(t, \xi, 1 : m)$ gives us the delay-vector for the (one-dimensional) input variable ξ at time t, but $\hat{\mathbf{X}}(t, \xi, 1 : m-2)$ gives a lower-dimensional reconstructed space. Anyhow, the tensor representation and the indicator variables, which are not merely scalars, but can be, as we have illustrated, arrays of values, allow us to define multiple features in different spaces using one single phase space reconstruction. Consequently, we randomly sample binary tests as follows:

$$f_\theta(t_1, t_2, \xi_1, \xi_2, \chi, \hat{\mathbf{X}}) = \|\hat{\mathbf{X}}(t_2, \xi_2, \chi) - \hat{\mathbf{X}}(t_1, \xi_1, \chi)\|_2 \tag{4.15}$$

where t_1, t_2 are time offsets constrained to be within the input patch.

The proposed test exploits the simplest form of relational features [MRC05]: distances between feature points or delay-vectors. However, since the features might be taken from the phase space, we are in fact relating the dynamics of body parts or features.

The proposed test is a very generic form of the binary tests employed for random forests in the phase space. In practice, we do not explore all the combinations of χ. We limit our random forests to sample from the input space ($\chi = 1$) and the embedded space ($\chi = 1 : m$). Also, ξ is conditionally dependent on the input space and on the value of χ. For instance, if we deal with joint positions in $\mathbb{R}^3$ and given $\chi = 1$, values of ξ are constrained to be multiples of 3, the input space dimensionality.

Note also that finding optimal embedding delays τ_ξ may provide, to some extent, temporal invariance within the extracted patches. Although time indices t_1 and t_2 are not subject to any temporal invariant transformation, delay-vectors contain temporally equispaced coordinates according to τ_ξ. Consequently, the Euclidean distance between two delay-vectors will give an idea of the similarity of dynamics almost independently of the speed of execution.

In test time, a set of patches, with a given length L frames and centered at a given time instant t are provided to the forest. It has been shown [SJC08, GYR+11] that dense sampling of test patches yields more accuracy. A common practice is to measure the density in terms of patch overlap. Consequently, test patches must sampled at rather close time instants in order to guarantee a sufficient overlap. As a rule of thumb, we will sample at intervals of $(L-1)/2$ frames. After testing down each patch, the random forest casts a vote at time t for each action class. These votes have an associated class probability. Class probabilities are aggregated using a Parzen estimator with Gaussian kernel K. For each action class, we perform the following aggregation:

$$p(y|\hat{\mathbf{X}}) = \sum_s p(y|\hat{\mathbf{X}}, t_s) K(t - t_s) \tag{4.16}$$

The result of this aggregation is depicted in Figure 4.6 for an example where 5 action classes are considered. Note that the random forests casts votes in the phase-space, and not in the input space, hence the temporal dimension (represented by the width of the image in Fig. 4.6) has $T - (m-1)\tau_{max}$ samples, where T is the length of the test motion sequence and τ_{max} is the larger embedding delay found in the input time series corresponding to the test motion sequence.

4.6 Experimental Results

In order to evaluate our approach to action recognition, in this section we conduct several experiments and we report the obtained results. This section is structured as follows. First, we conduct experiments on a publicly available motion capture dataset.

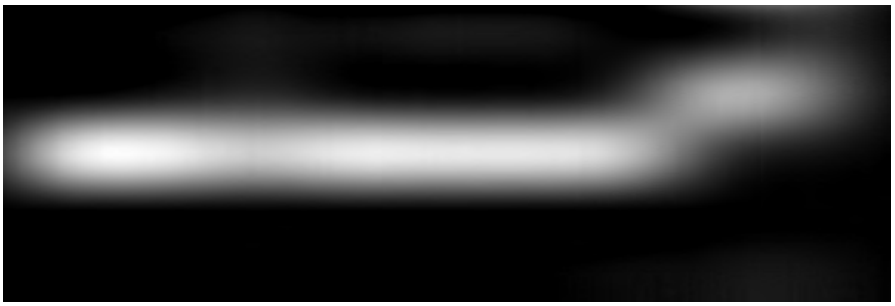


FIGURE 4.6: Aggregation of votes in the random forest framework. In this example, 5 action classes are considered. The width of the image represents the temporal dimension, whereas the height is divided into 5 intervals where the aggregated votes for each class are represented. At the beginning of the sequence, the forest casts votes for class 3. By the end, class 2 is voted. Note: Blurring along vertical axis is caused during the upsampling performed for visualization purposes.

In these experiments we analyze actions employing articulated models, which is the core of this dissertation. Next, we present experimental results on a publicly available video dataset. Despite being out of the scope of this thesis, we aim to show that the proposed framework works also for data not coming from articulated models. Finally, we show results on a multi-view video dataset, where we combine pose inference and the proposed action recognition framework.

4.6.1 Future Light Dataset

The motion capture dataset provided by FutureLight (FL) [Fut] contains a total of 158 sequences of 5 action classes: *dance*, *jump*, *run*, *sit* and *walk*. This is a challenging dataset for two reasons. First, actions have important intra-class variability. For instance, *run* and *walk* actions are performed at different speeds, present variations in arm swinging, and include stops and turnings. Similarly, *jump* class sequences involve jumping in place or while walking. Second, the dataset contains somewhat ambiguous action instances. As an example, *dance* sequences contain 4 walk-style dance moves that are similar to walking. A good description of this dataset, illustrated with some samples, can be found in [ABS07].

Analysis of Articulated Model Variables As described in Section 4.3, we can use joint angles and joint positions in order to construct a composite feature vector for human action recognition. Such a feature vector will be in general of high dimensionality and some of the methods presented in this dissertation might not effectively handle this dimensionality. For that matter, and in order to assess the information encoded by several features, we conduct experiments measuring the importance of features obtained in an articulated human body model. Let us recall that, in any case, we will represent

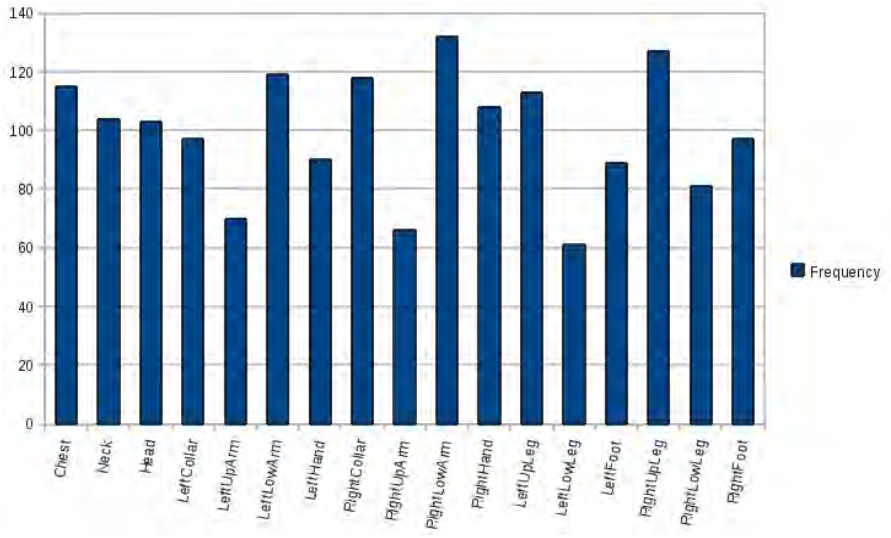


FIGURE 4.7: Frequency (number of times) of selection of joint angles.

joint positions in a view-invariant manner, according to what we exposed in Section 4.3. That is, before computing the joint positions, we discard the global orientation and the translation in the XY plane.

To objectively measure the importance of joint angles and joint positions, we employ the random forest approach presented in Section 4.5. Note that the objective of this experiment is not to evaluate the classification accuracy, but to measure which features are picked more frequently. For that matter, we train a random forests of 10 trees and maximum depth 15. We employ 500 random binary tests and, for each test, we sample 10 random thresholds. Tests are constrained to pick whether joint angles or joint positions. Embeddings are discarded for the moment. Their relevance for action recognition is evaluated experimentally later on. Having the trained forest, we analyze which features have been selected more frequently.

Our results show that joint angles are picked more frequently, although it is not a vast majority of times. Specifically, joint angles are picked in 55% of the branching nodes. Figures 4.7 and 4.8 show the number of times each joint is selected when it is represented in terms of rotations or as a 3D position. Provided that joint angles are picked more often, we analyze each axis separately (see Fig. 4.9) The results provided by this analysis show that, in general, most of the relevance is carried by a single rotation axis.

Trajectory Matching Based on the previously presented results on articulated model variable selection, we employ a total of 27 variables including relevant joint rotations from 8 joints from legs and and arms (according to Fig. 4.7), 2 global rotations and normalized translation in the vertical axis. With this data we obtain 27 reconstructed

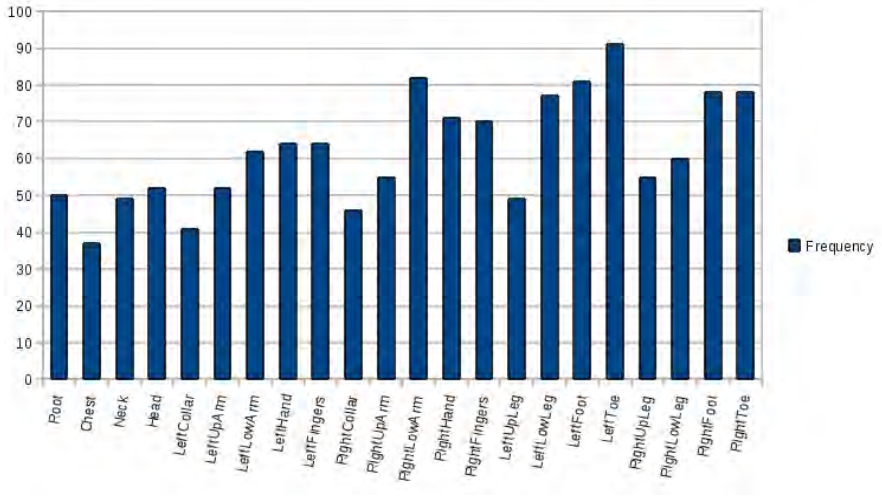


FIGURE 4.8: Frequency (number of times) of selection of joint positions.

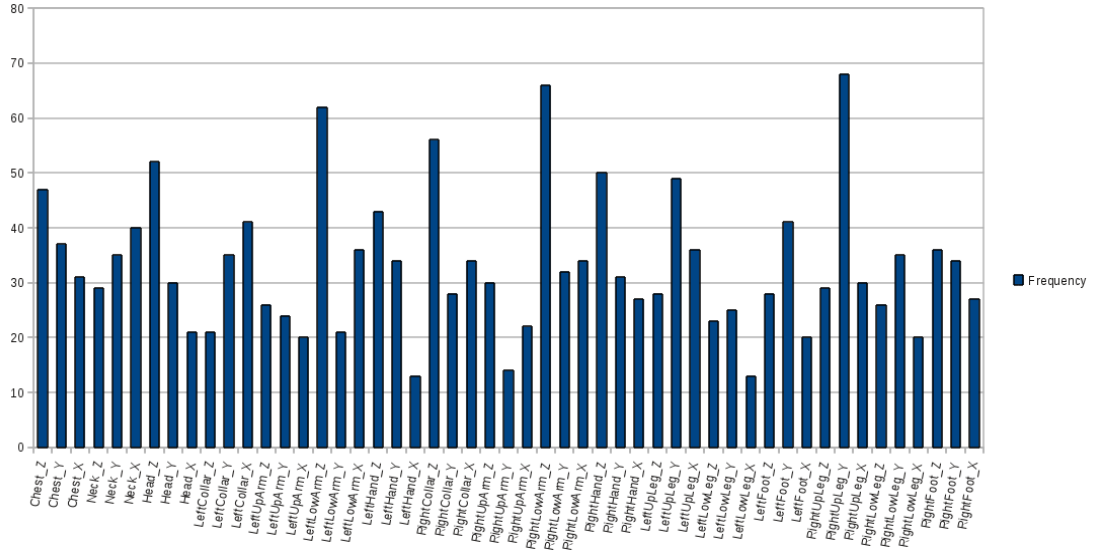


FIGURE 4.9: Frequency (number of times) of selection of angular features. Axis rotations are depicted separately.

spaces per action sequence. Classification performance is evaluated using leave-one-out cross-validation, as has been done by other authors working in action recognition with this dataset [ABS07, RWS10, RWS08]. We report the overall classification accuracy (in terms of number of correctly classified sequences out of the total number of sequences) and the average classification accuracy (in terms of the average classification accuracy per class). Provided that in the Future Light classes are unbalanced (*jump* has 14 sequences while *walk* has 48), we should only provide average accuracy over classes. However, some authors reported the overall performance, hence we include both metrics. Confusion matrices and classification results are shown in Table 4.1. Overall and average classification accuracies (93.08% and 88.67% for the SVM classification and 93.67% and 91.09% for the weighted approach) validate the performance of reconstructed phase

	Dance	Jump	Run	Sit	Walk
Dance	**29**	2	0	0	0
Jump	0	**11**	1	0	2
Run	1	1	**25**	0	3
Sit	0	0	0	**35**	0
Walk	0	0	0	0	**48**

	Dance	Jump	Run	Sit	Walk
Dance	**28**	2	0	1	0
Jump	0	**8**	4	0	2
Run	0	1	**28**	0	1
Sit	0	0	0	**35**	0
Walk	0	0	0	0	**48**

TABLE 4.1: Confusion Matrices in the Future Light dataset for the Trajectory Matching in Phase Spaces. Left: weighted approach (overall and average classification accuracies are 93.67% and 91.09% respectively). Right: SVM approach (overall and average classification accuracies are 93.08% and 88.76% respectively). In both cases the embedding dimension $m = 5$ and δ=10.

portraits and the proposed distance for the action recognition task. The comparison between the weighted and SVM classification approaches reveals that the latter is prone to overfitting. This overfitting is caused by the unbalance between class *jump* and other classes. Provided that the proposed approach deals with each sequence as an example and therefore class *jump*, having a reduced number of examples, obtains a poor classification accuracy. This clearly reduces the average classification accuracy on this dataset.

Using the same model variables, we evaluate the recognition rate as a function of the embedding dimension (Fig. 4.10). In the case of $m = 1$ we perform DTW [BC96] using a window to constrain the warping to near time instants of the evaluated sample. In this way, the computation time is dramatically reduced. As expected, the recognition rate generally improves as the embedding dimension m grows, although we obtain a surprisingly high accuracy by just employing $m = 2$. If we look at the E1 measure computed using the Cao's false nearest neighbor method on the overall FL dataset, we see that at $m = 5$ the E1 measure starts settling down to a convergence value, whereas the slope is reduced by $m = 3$. This average behavior is correlated with the overall recognition performance. A remarkable increase of E1 (i.e., reductions of false nearest neighbors) usually matches an increase in the accuracy. For embedding dimensions larger than 7, the finite duration of sequences becomes a practical limitation, and the accuracy decays.

The proposed action recognition system is mainly implemented in MATLAB, except for some routines that have been implemented in C/C++. In the Future Light dataset, action recognition (in test time) takes an average of 500 ms per mocap frame in an Intel 2.4GHz CPU with 4GB of RAM (both SVM and weighted strategies take approximately the same time, since most of the computational effort is devoted to computing the distances between phase portraits). However, our approach can be further optimized by computing distances to templates in parallel or using a tree efficient organization of the training trajectories.

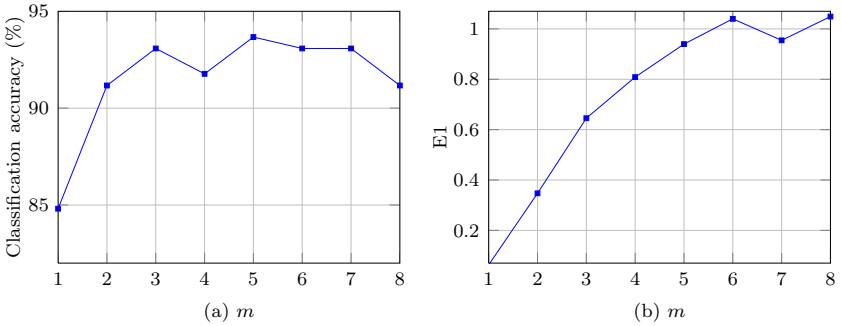
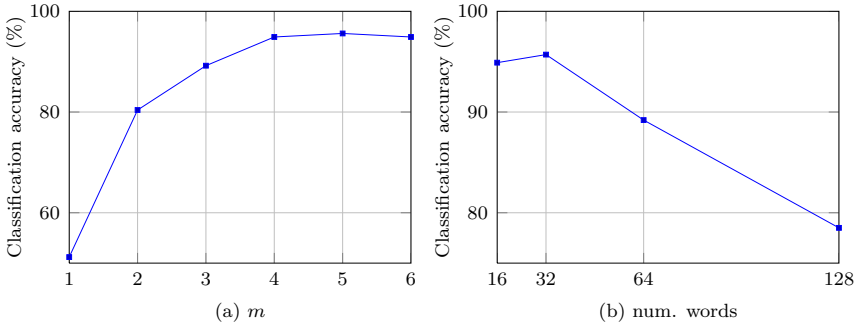


FIGURE 4.10: Analysis of the overall recognition rate and the embedding dimensions in the Future Light dataset (a) Recognition rate as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the whole Future Light dataset.

BoDV As in the trajectory matching experiments, and based on the analysis of articulated model variables, we select a subset of 27 skeleton variables, thus obtaining 27 reconstructed spaces per action sequence.

In the experiments, we test several parameters such as embedding dimensions and number of words. Classification performance is evaluated using leave-one-out cross-validation, as reported in previous approaches [ABS07][RWS10][RWS08]. We also compare k-means and hierarchical clustering approaches withing the BoDV framework. The hierarchical clustering implementation used in this thesis employs a pre-computation of the distances between data points in order to speed-up the clustering. Since this implementation requires a considerable amount of memory, we downsample the time-series to 4:1 ratio. Provided that mocap data is recorded at 120Hz and that consecutive samples are strongly correlated, the downsampling should not considerably affect the performance of the BoDV.

Confusion matrices and overall and average recognition accuracy (see experiments with trajectory matching for details) are reported in Table 4.2. We also show the performance of the BoDV model with hierarchical clustering for several embedding dimensions and number or words (Fig. 4.11). We report an average recognition performance of 94.05% by embedding the time series in a 5-dimensional space, 32 words and optimal delay embedding provided by the mutual information method. Comparative results between clustering methods show that hierarchical clustering outperforms k-means thanks to a more accurate geometric characterization (achieved even at lower frame rates). We also observe that the number of words is a critical parameter for BoDV. A large number of words yields a bad performance. This behavior was expected to some extent, since we deal with time series of each model variable isolatedly, and in the Future Light dataset there are a total of 5 action classes and 158 examples. Consequently, and since we stack the histograms obtained in the reconstructed spaces of each of this variables, the feature vectors are of a huge dimensionality if the number of words is relatively high. Apart



(a) m (b) num. words

FIGURE 4.11: Overall classification results in the FL dataset with BoDV and hierarchical clustering (a) for a fixed number of words (32) and varying embedding dimension and (b) for a fixed embedding dimension ($m = 5$) and varying number of words.

	Dance	Jump	Run	Sit	Walk
Dance	**28**	1	0	0	2
Jump	0	**10**	2	0	2
Run	0	0	**28**	0	2
Sit	0	0	0	**35**	0
Walk	0	0	0	0	**48**

	Dance	Jump	Run	Sit	Walk
Dance	**31**	0	0	0	0
Jump	1	**11**	1	0	1
Run	0	0	**30**	0	0
Sit	0	0	0	**35**	0
Walk	0	0	3	0	**44**

TABLE 4.2: Confusion Matrices in the Future Light dataset for the BoDV approach. Left: k-means clustering with $m = 5$ and $nw = 16$ (overall and average classification accuracies are 94.30% and 91.02% respectively). Right: hierarchical clustering approach with $m = 5$ and $nw = 32$ (overall and average classification accuracies are 95.57% and 94.05% respectively). In both cases we present the parameter set providing the best accuracy.

from dimensionality issues, we observe that we do not even reach 32 words in some cases where the time series are embedded in compact sets of delay-vectors. This means that forcing a large number of words produces not only a very high dimensional feature space, but also unnecessarily sparse feature vectors for which learning becomes hard.

Similarly to trajectory matching results presented previously, the overall classification accuracy is correlated with the E1 measure of the overall Future Light dataset. In this case, we only show the performance for dimensions 1 to 6, since larger embedding dimensions are not practical due to finite length of the time series, and result in a worse performance.

The BoDV action recognition approach is mainly implemented in MATLAB, except for some routines that have been implemented in C/C++. Action recognition (in test time) takes an average of 1.3 ms per mocap frame in an Intel 2.4GHz CPU with 4GB of RAM (similarly to trajectory matching, the two quantization methods required by the k-means or hierarchical clustering approach require approximately the same time). Hence, compared to trajectory matching, BoDV is two orders of magnitude faster in test time, reaching real-time performance.

Random Forests in the Phase Space Differently to the previous approaches, random forests can efficiently handle high dimensional spaces. In addition, despite the relatively low number of sequences of the Future Light dataset, the proposed random forest approach works locally with patches of a given length and consequently, we can randomly sample a number of patches from each training sequence. For these reasons, the binary tests proposed in Equation 4.15 are sampled using the following combinations of randomly sampled indicator variables:

- A number of joint angles in $\mathbb{R}$ can be used for computing the binary test setting $\chi = 1$ and sampling $\xi \sim \mathcal{U}_{[1,K]}$, where $\mathcal{U}$ is a uniform probability distribution between 1 and K, the number of joint angles considered for action recognition.
- A number of joint positions in $\mathbb{R}^3$ can be used in the binary tests by setting $\chi = 1$ and using $\xi = K + 1 + 3\xi'$ where $\xi' \sim \mathcal{U}_{[0,P-1]}$. Here P is the number of joint positions of the articulated model.
- Binary tests using the phase space reconstructed from the multivariate time series comprising joint angles and positions. These embedded features are obtained by setting $\chi = 1 : m$ and by $\xi \sim \mathcal{U}_{[K+3P+1,\Xi]}$ (recall that Ξ denotes the number of univariate time-series resulting from the articulated model).

In order to randomly sample the binary tests according to the described combinations, we define a new integer indicator variable ν uniformly distributed between 1 and 3. Its objective is to select between the three presented modes, prior to sampling from any of them.

Using the described random tests, we train forests of 10 trees and maximum depth 10. We employ patches of 25 frames. These patches are randomly sampled from the available training sequences. As in the previous experiments, we follow a leave-one-out cross-validation approach. Following the rule of thumb described in Section 4.5, we sample test patches at every 12th frame. Each patch is tested with the learned forest and votes with some associated probability are casted. Class probabilities are aggregated using a Parzen estimator with a Gaussian kernel. A sequence is then labeled with the action class having the larger probability in the majority of the available frames. If two or more actions have the same number of frames with maximum probability, then the sum of the per frame probabilities is taken into account in order to decide for one single class. Recall that, the number of frames considered in test time is that of the embedded data $\hat{\mathbf{X}}$ (see Section 4.5).

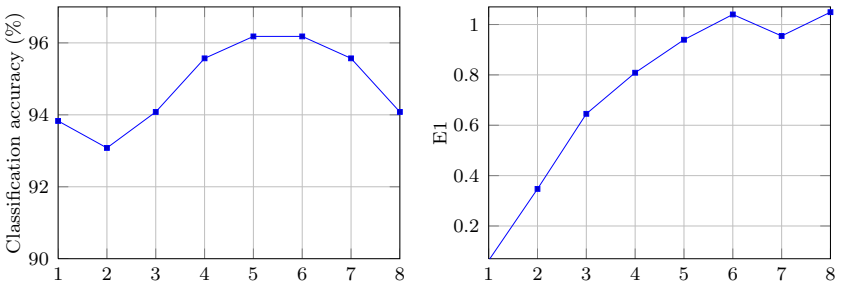
Using a common embedding dimension $m = 5$, we report and overall recognition accuracy 96.83% and an average recognition accuracy of 96.18% (see experiments with

	Dance	Jump	Run	Sit	Walk
Dance	**30**	0	0	0	1
Jump	0	**13**	1	0	0
Run	0	0	**28**	0	2
Sit	0	0	0	**35**	0
Walk	0	0	1	0	**47**

TABLE 4.3: Confusion Matrices in the Future Light dataset for the random forest approach with $m = 5$. Overall and average classification accuracies are 96.83% and 96.18% respectively.

trajectory matching for details on these two metrics). The corresponding confusion matrix is reported in Table 4.3. The confusions are in general understandable due to the fact that some of the action sequences of this dataset are fairly ambiguous. Interestingly, in several of the reported confusions, our approach votes, at some point of the sequence, for the true class. For instance, in the confusion between *jump* (true action class) and *run* (predicted action class), 24% of the frames receive votes for the true action class *jump*. Similarly, in the misclassified *walk* sequence our forest periodically votes for the correct motion class.

We evaluate the random forest approach for different embedding dimensions. Results are shown in Fig. 4.12. Similarly to the previous approaches, we observe that an appropriate embedding dimension yields better performance. Independently of the classification approach and the features employed, the classification accuracy grows with the embedding dimension, until an upper bound is reached. This upper bound is mostly due to practical constraints: time series have a finite duration, hence one cannot indefinitely increase the embedding dimension without losing some information. Nevertheless, the most important outcome is that employing time-delay embeddings improves the performance over only using the input feature space. In fact, the random forest training reveals the importance of delay-vectors in the classification of actions. We observe that embedded features are picked 71% of the time, while angles and joint positions are chosen 17% and 12% of the time respectively. Using an embedding of $m = 5$, we provide an overview of the joints selected by the forest when working in the embedded space. We basically visit all the leafs from all the trees and we associate the joints that have been selected in the parent nodes. In this analysis, we do not make distinctions between rotations, positions or coordinates; we simply take into account if the joint has been selected in the binary test. Figures 4.13, 4.14, 4.15, 4.16 and 4.17 show the outcome of such an analysis. In general, no joint seems to be a very specific trait of any of the actions of the Future Light dataset. This is indeed caused by the locomotion segments present in all the actions of this dataset. For this reason, differences in the motion of specific joints tend to be the discriminative characteristic of actions. For instance, rotations and positions of the right hand are frequently selected to classify all the actions. This actually indicates that



(a) m (b) m

FIGURE 4.12: Analysis of the average recognition rate and the embedding dimensions in the Future Light dataset (a) Recognition rate of the random forest approach as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the whole Future Light dataset.

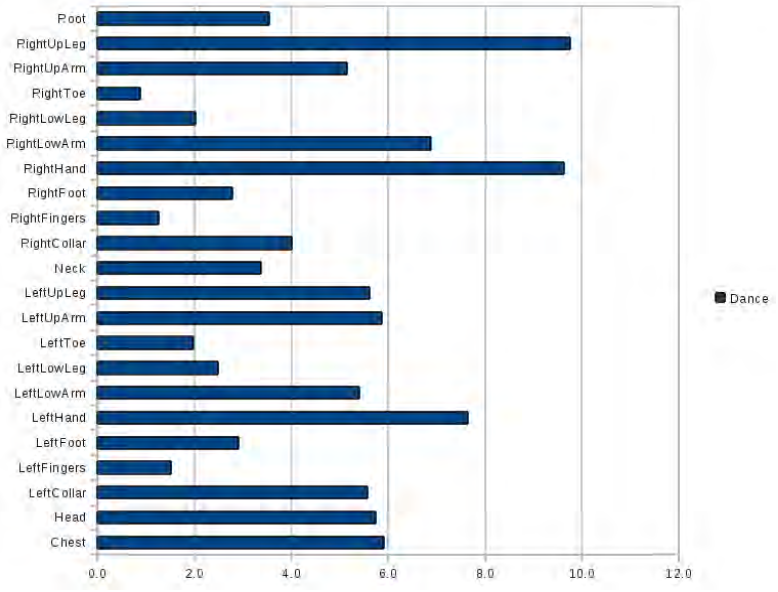


FIGURE 4.13: Frequency (% of times) of selection of joints for DANCE action.

the random forest finds distinctive motion patterns in the phase spaces associated to the right hand joint.

As far as computational performance is regarded, the proposed random forest approach is faster than the preceding methods. While time-delay embedding code is implemented in MATLAB, random forest are implemented in C++, with no optimization efforts. In test time, we estimate an average of 1 ms per mocap frame in an Intel 2.4GHz CPU with 4GB of RAM. Consequently, the random forest approach is faster and more accurate than the preceding methods.

Comparison to the State-of-the-Art We summarize the obtained results in comparison to existing state-of-the-art methods evaluated in the FutureLight dataset (see Fig. 4.18). In particular, we compare our methods with spike-train driven dynamical

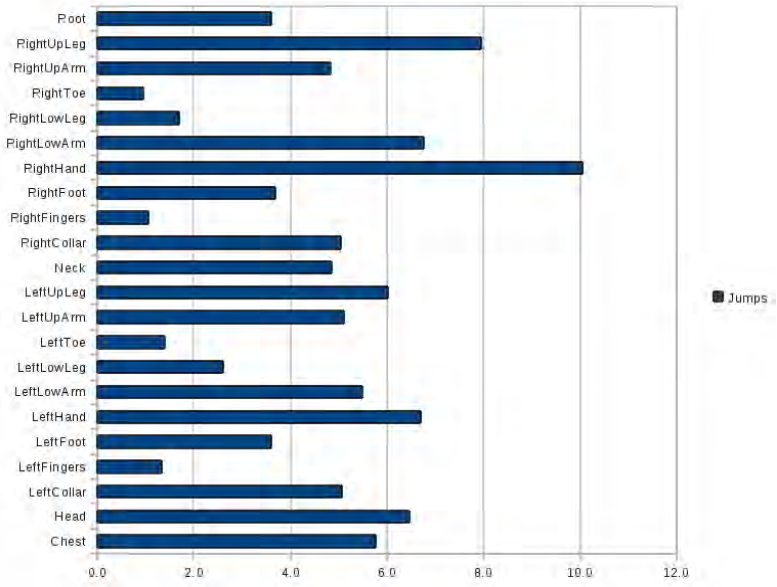


FIGURE 4.14: Frequency (% of times) of selection of joints for JUMP action.

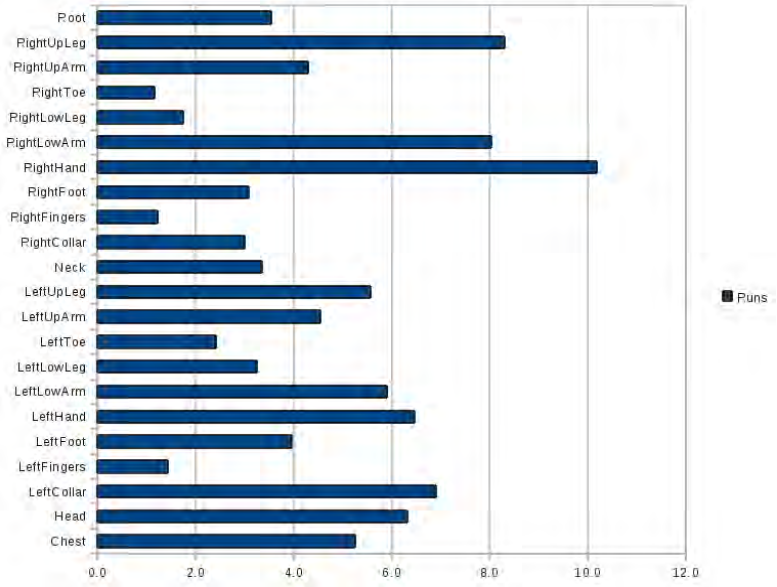


FIGURE 4.15: Frequency (% of times) of selection of joints for RUNS action.

models [RWS10], chaotic invariants [ABS07] and flexible dictionaries for action classification [RWS08]. Note that the results reported in [ABS07] use only 155 sequences of the FL dataset, hence overall and average classification results might not be directly comparable. The methods proposed in this dissertation outperform most of the existing approaches [RWS10, ABS07]. Compared to our random forests, only the flexible dictionaries method by Raptis et al. [RWS08] delivers a slightly better performance in the FutureLight dataset. This slightly improved performance is due to robustness against the strong intra-class variations present in the FutureLight dataset. As stated at the beginning of this section, the intra-class variations of this dataset yield, in some cases,

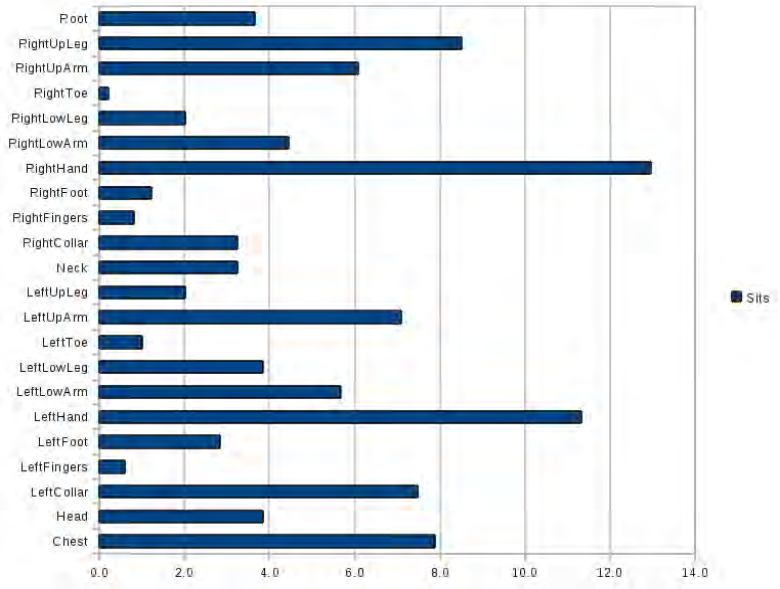


FIGURE 4.16: Frequency (% of times) of selection of joints for SIT action.

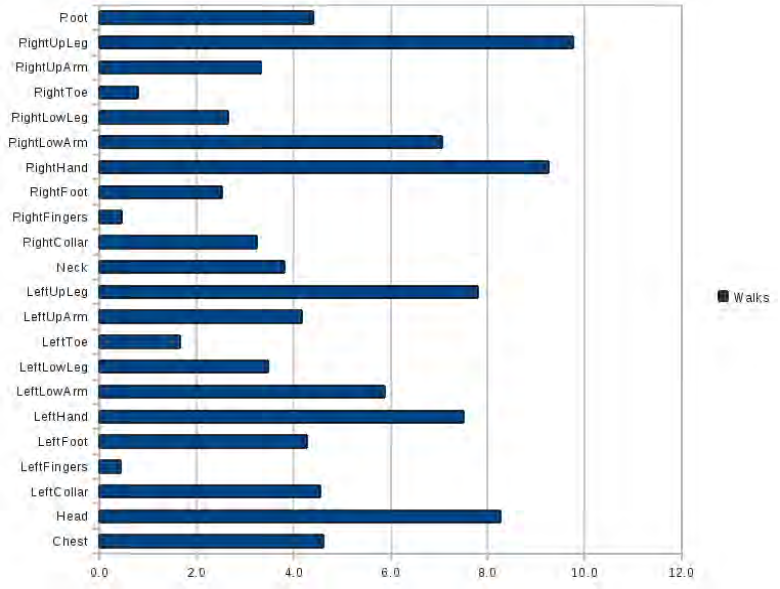


FIGURE 4.17: Frequency (% of times) of selection of joints for WALK action.

fairly ambiguous sequences. This ambiguity makes some of the misclassifications of our approach understandable, as for instance, one of the misclassified *jump* sequences starts by the actor running in place (and our approach votes for *run* at the beginning of this sequence). Another important characteristic of our approach, in comparison to [RWS08], is that we do not require numerical derivatives (which might be prone to errors in the presence of noise) and we deliver a competitive performance even at a lower frame rate. Notice that, in addition, the time-delay embedding framework can be also used in order to synthesize motion [BS09].

Method	FL Overall	FL Average
Spike Train Driven Dynamical Models [RWS10]	83.63%	86.07%
Chaotic Invariants [ABS07]	89.68%	90.20%
Flexible dictionaries + BoW [RWS08]	97.47%	97.40%
Flexible dictionaries + String Matching [RWS08]	98.10%	98.03%
Trajectory Matching (SVM Approach)	93.08%	88.76%
Trajectory Matching (Weighted Approach)	93.67%	91.09%
BoDV (k-means clustering, $m = 5$, 16 words)	94.30%	91.02%
BoDV (hierarchical clustering, $m = 5$, 32 words)	95.57%	94.05%
Random Forest in Phase Space ($m = 5$)	96.83%	96.18%

FIGURE 4.18: Comparative results for the FutureLight dataset. Overall and average classification accuracies are reported

Trajectory matching is most negatively affected by such intra-class variations. This approach makes a comparison that is rather exhaustive, in the sense that all points of the available trajectories in the phase-space are matched. Therefore, our trajectory matching method presents limitations in coping with significant performance differences between the same action class. However, this approach presents the advantage of working with a relatively reduced number of training examples in comparison to the codebook based approaches. In Section 4.6.3 we conduct several experiments with markerless motion capture that evaluate the reliability against reduced number of training samples.

The comparative performance of the method by Ali et al. [ABS07] and our methods is relevant, since they use chaotic invariants and time-delay embeddings. The accuracy of our algorithms on this data is consistently better. On the one hand, and considering the accuracy reported in [ABS07], this proves that methods using time-delay embeddings perform well in recognizing human actions from MoCap data. On the other hand, we show that geometric and topological analysis on a common embedding or phase-space consistently outperform the chaotic invariants analysis when using MoCap data, and in particular, allow using powerful classification methods, such as random forests, in order to provide a much improved action recognition performance. Finally, apart from the improved recognition performance, avoiding chaotic invariants allows our method to perform well for short term actions, as we show in experiments with video data (see next Section).

4.6.2 Weizmann Dataset

The Weizmann dataset [GBS+07] consists of 90 low-resolution videos of nine actors performing 10 different actions: *bend*, *jack* (jumping jack), *jump* (jumping forward on two legs), *pjump* (jumping in-place on two legs), *skip*, *side*(galloping sideways), *run*, *walk*, *wave1* (wave one hand) and *wave2* (waving two hands). These videos are recorded from the same viewpoint. The dataset also provides background subtracted human silhouettes

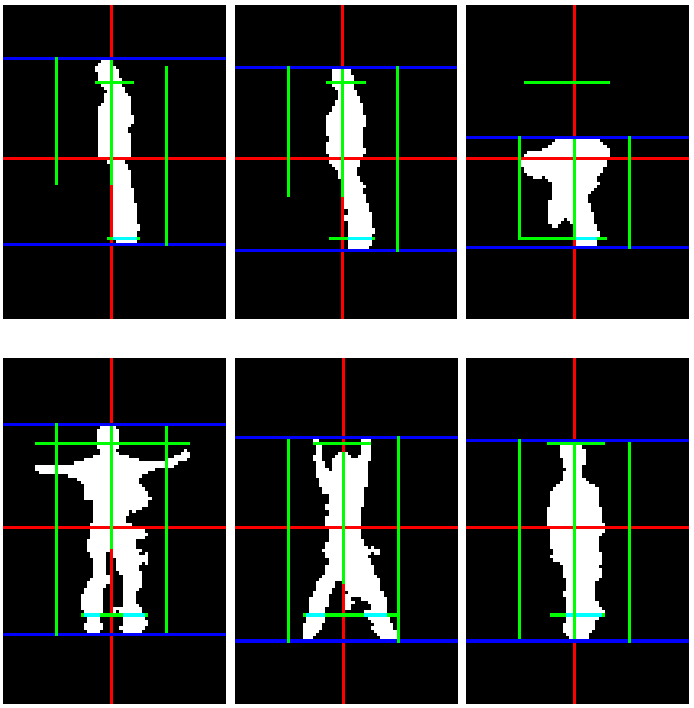


FIGURE 4.19: Features obtained in the Weizmann dataset (best viewed in color). Red lines represent the quadrants. Green lines depict the width and height of the silhouette in each quadrant, as well as the medial height. Blue lines are the top and bottom of the silhouette, used to compute the silhouette height.

in order to perform action recognition. The evaluation protocol mostly employed on this dataset is leave-one-out cross-validation.

Although state-of-the-art approaches have saturated the performance on this dataset, it is still a popular benchmark dataset often included in action recognition publications. Moreover, the chaotic invariants approach proposed in [ABS07] is evaluated on this video dataset. This fact makes the evaluation on the Weizmann dataset relevant for comparison against an approach already relying on time-delay embeddings.

Features We propose to use a set of simple features similar to the ones proposed in [RWS08] for the same dataset. We start by aligning the silhouettes such that the centroids occupy the same pixel in the middle of the image. In order to compute some pose-like variables, we divide the aligned foreground silhouette into 4 quadrants and we compute the width and height of the foreground pixels within each quadrant. Additionally, we compute the silhouette height and width, and the medial height, i.e., distance between top and bottom foreground points along the the vertical axis. In total, we have 11 features that approximately represent the human pose in the given view (see Fig. 4.19).

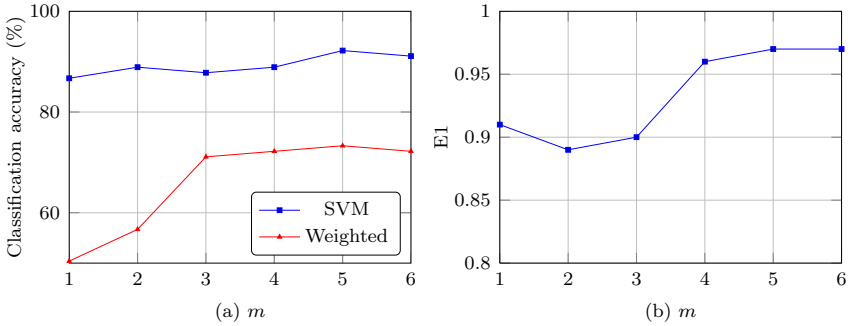


FIGURE 4.20: Analysis of the average (per class) recognition rate and the embedding dimensions in the Weizmann dataset (a) Recognition rate as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the Weizmann dataset.

Trajectory Matching We use the described features and their associated time series in order to reconstruct 11 phase spaces. We first compute the minimum common embedding dimension. According to the criteria presented in Section 4.4.2, we find that a dimension of $m = 4$ suffices in order to recognize actions in the phase space (see Fig. 4.20b). We then apply the SVM and weighted trajectory matching approaches with parameter $\delta = 5$. As in the experimental results with mocap data, we verify the impact of the embedding dimension in the recognition accuracy. The classification accuracy as a function of the embedding dimension is shown in Figure 4.20. While the SVM approach performs well, with an average recognition accuracy of 92.2% for $m = 4$, the weighted approach fails to fuse the importance of each variable and delivers a poor 73.3% of average recognition accuracy. Our hypothesis around this poor performance is that weighted learning approach is rather simple method that builds on the concept of body parts, which in the Future Light dataset can be approximately represented by joint rotations. In the Weizmann dataset, the proposed features do not represent body parts as precisely as joint rotations do and that is a caveat for the weighted approach. Confusion matrices for both trajectory matching approaches are shown in Fig. 4.21. The reason of such a poor behavior is found on the weak reliability of a 1-NN classifier on this dataset. Recall that the weight of each variable is learned from a 1-NN classifier, and that the classification is based also on a weighted 1-NN.

Bag of Delay-Vectors The delay-vectors lying in each of the 11 reconstructed phase-spaces are clustered in order to compute the words for the BoDV approach. As in the case of mocap data, we evaluate several embedding dimensions, several number of words and the two proposed clustering schemes. Results for the best working number of words ($nw = 16$) are depicted in Figure 4.22. We report an average classification accuracy of 88.9% for the BoDV with $nw = 16$, $m = 3$ and hierarchical clustering (confusion matrices are shown in Fig. 4.23).

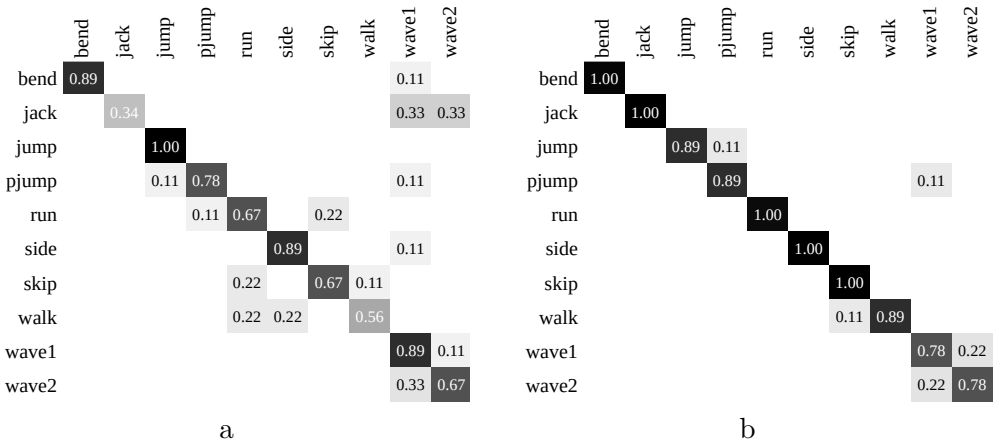


FIGURE 4.21: Confusion matrices in the Weizmann dataset for (a) trajectory matching and weighted approach (b) trajectory matching and SVM approach.

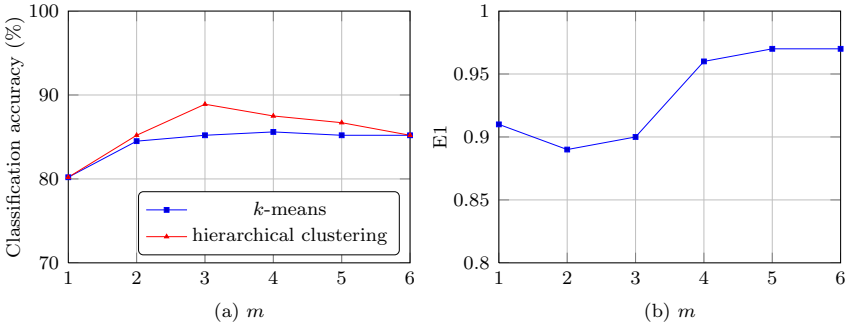


FIGURE 4.22: Analysis of the average (per class) recognition rate and the embedding dimensions in the Weizmann dataset (a) Recognition rate as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the Weizmann dataset.

The lower performance obtained by BoDV in this dataset is attributed to two main reasons. Firstly, the nature of the input features, which are not real pose features, but approximations. Secondly, the shorter length of the available sequences, which yield a substantially lower number of samples to create the BoDV models for action recognition. This second reason is supported by the fact that the best performance is achieved with a lower embedding dimension (a lower dimension implies more delay-vectors). These reasons indicate that BoDV, which only rely on the geometry of the reconstructed phase-space, do not generalize as well as trajectory matching.

Random Forests in the Phase Space Similarly to the mocap data experiments, the binary tests proposed are sampled using combinations of randomly sampled indicator variables. In the case of the Weizmann dataset, we propose the following:

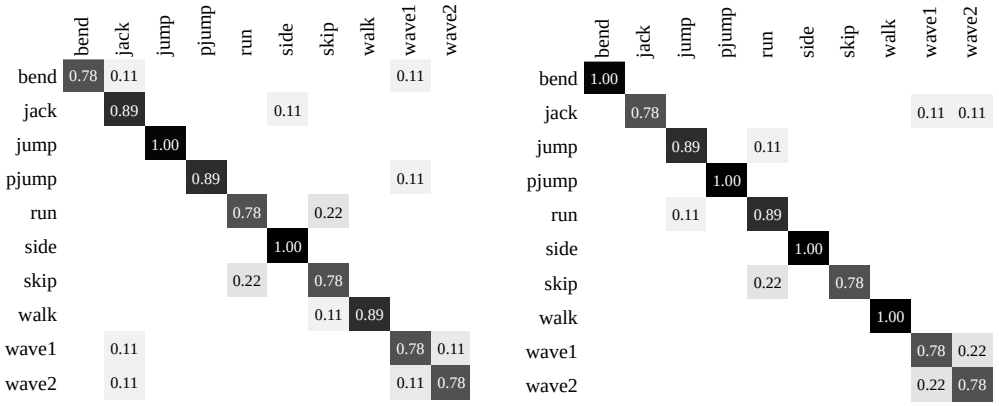


FIGURE 4.23: Confusion matrices in the Weizmann dataset for (a) BoDV with k-means (b) BoDV with hierarchical clustering

- Input features in $\mathbb{R}$ can be used for computing the binary test by setting $\chi = 1$ and sampling $\xi \sim \mathcal{U}_{[1,K]}$, where $\mathcal{U}$ is a uniform probability distribution between 1 and $K = 11$, the number of input features considered for action recognition.
- Binary tests using the phase space reconstructed from the multivariate time series comprising the input features. These embedded features are obtained by setting $\chi = 1 : m$ and by sampling $\xi \sim \mathcal{U}_{[K+1,\Xi]}$.

We define the integer indicator variable ν uniformly distributed between 1 and 2, in order to randomly select one of the presented modalities.

Using the described random tests, we train forests of 10 trees and maximum depth 8. We employ patches of 5 frames. These patches are randomly sampled from the available training sequences. According to Section 4.5, we sample test patches every 2 frames. Each patch is tested with the learned forest and casted votes are aggregated. A sequence is then labeled with the action class having the larger probability in the majority of the available frames. As in the case of mocap data, if two or more actions have the same number of frames with maximum probability, then the sum of the per frame probabilities is taken into account in order to decide for one single class.

Results as a function of the embedding dimension are depicted in Fig. 4.24. We report an accuracy of 97.8% for $m = 4$. The confusion matrix obtained in that case is shown in Fig. 4.25.

Comparison to the State-of-the-Art We compare the obtained results with existing state-of-the-art approaches working on the Weizmann dataset. We compare the results in terms of mean per class accuracy, i.e., the average of classification accuracies

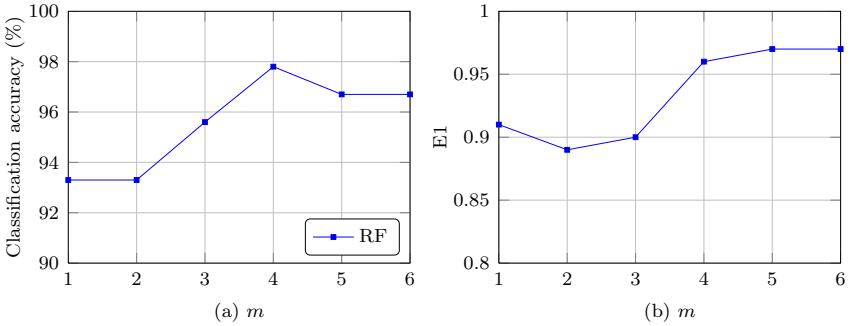


FIGURE 4.24: Analysis of the average (per class) recognition rate and the embedding dimensions in the Weizmann dataset (a) Recognition rate as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the Weizmann dataset.

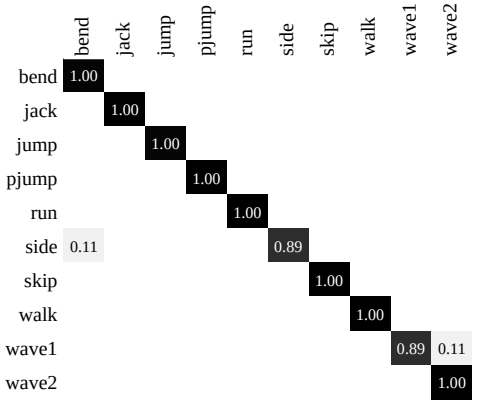


FIGURE 4.25: Confusion matrix in the Weizmann dataset for the random forest approach.

in each class (see Fig. 4.26). All the reported state-of-the-art results have been obtained from the authors' original publications. However, some approaches present some minor differences in the evaluation methodology. Specifically, half of the approaches are tested on a previous version of the Weizmann dataset comprising 9 actions (*skip* is missing in this reduced set).

Our random forest approach in the phase space delivers state-of-the-art performance, even when the employed features are rather simplistic. We obtain the same accuracy that [GYR+11], despite their Hough forest is trained and tested with ground truth tracks. If these ground truth tracks are not used in test time, they report a 95.6% accuracy on this dataset. The comparison with [ABS07] and [WOS11] is, as in the case of mocap data, particularly interesting, because they use chaotic invariants computed from time-delay embedded time series. While the BoDV and trajectory matching approaches are outperformed by chaotic invariants in this dataset, we obtain a 5% improvement with the random forest approach.

[1]Results reported by authors using the 9 action Weizmann dataset

Method	Accuracy
Chaotic Invariants [ABS07]	92.6% [1]
Chaotic Invariants + Trajectories [WOS11]	92.8% [1]
Flexible dictionaries + BoW [RWS08]	95.1% [1]
Flexible dictionaries + String Matching [RWS08]	100% [1]
Bag of Snippets [SvG08]	100% [1]
Ommer et al. [OMB09]	97.2% [1]
Prototype Tree [JLD12]	100%
Hough Forest [GYR+11]	97.8%
Trajectory Matching (SVM Approach)	92.2%
Trajectory Matching (Weighted Approach)	73.3%
BoDV (k-means clustering, $m = 4$, 16 words)	85.6%
BoDV (hierarchical clustering, $m = 3$, 16 words)	88.9%
Random Forest in Phase Space ($m = 4$)	97.8%

FIGURE 4.26: Comparative results for the Weizmann dataset. Average classification accuracies per class are reported

4.6.3 Multi-view Video dataset

Finally, we present results on a more realistic scenario, where we combine pose estimation and action recognition. To this end, we record several sequences involving four actors performing several actions. This test scenario is intended to simulate a smart environment where each user interacts with the room through 4 calibrated cameras located at the ceiling corners (see Fig. 4.27). In this context, intra-class variations of the actions performed are not as relevant as in the case of the Future Light dataset, because in this dataset many actions serve as commands, and are executed in a somewhat precise manner. In spite of that, some stylistic variations exist, especially between actors. We employ 7 minutes of video data containing the following actions: *walk*, *raise hand*, *crouch*, *wave hand*, *bounce* (an arm movement similar to bouncing a ball), *jump*, *clap*, *kick*, *sit* and *stand up*. Actions involving a single arm or leg motion are performed with the right limb. Except for walk, all the actions last a few seconds (typically between 1 and 4 seconds). Short observations make difficult to robustly estimate the optimal delay embedding by means of the mutual information. Even if one can find a suitable embedding delay such that time series get effectively unfolded in the phase-space, the limited duration of the actions will produce reconstructed phase portraits with a few points in several cases, making unfeasible to apply chaotic invariants reliably unless some re-sampling is performed [ABS07]. Concatenation of sequences has been proposed also as a possible solution, but in some actions concatenation may produce jumps that would violate the continuity of the dynamics. The metric distance that we propose in this paper does not require a very good quality of the reconstructed phase portraits, since we are analyzing the trajectories in the phase space. Since the estimated time delays and the embedding dimensions allow having at least 15 reconstructed points in the worst case, we do not perform any up-sampling nor concatenation.

FIGURE 4.27: Multi-view data-set samples. Top row: The four available viewpoints during walk action. Bottom row: Kick, jump, sit and bounce samples.

To track the actors, we employ a Layered Particle Filter with hierarchical layers (see Chapter 3). Although the state vector strictly comprises pose and anthropometric variables, we simplify the problem by computing anthropometric variables only once at the beginning of each subject's sequence [ACP10]. The variables in the pose vector are split into 7 layers, each of which represents global and torso variables, arms and legs. Each layer employs 500 particles.

Once the pose is estimated, we use 19 variables, since our model is simpler than the one resulting from marker-based motion capture in [Fut]. We estimate an average tracking error of 15 cm for all the considered data, although errors occur especially in some arm motions (clapping), in fast motion (kick, jump) and in crouching actions, where the tracker makes important errors in estimating the legs' pose. In spite of that, the tracking provides an approximate representation of the motion involved in each action (see Fig. 4.28).

For evaluation, we employ a total of 192 tracking instances from all the action classes (8 are discarded due to tracking failures caused mainly by foreground segmentation errors). We split the data into two sets containing 2/3 and 1/3 of the tracking instances, which are used for training and testing respectively. We repeat this procedure 10 times to obtain the average performance.

Trajectory Matching To evaluate the classification performance, we divide the data into several training and testing sets. Training sets contain 2/3 of the available data while test sets contain the remaining 1/3. We repeat this procedure 10 times with random partitions of the data for both the SVM and weighted approaches. Using these training sets we find that $m = 3$ is the minimum common embedding dimension. In all the experiments we use parameter $\delta = 5$. The classification results as a function of the embedding dimension are shown in Fig. 4.29. We report an average accuracy of

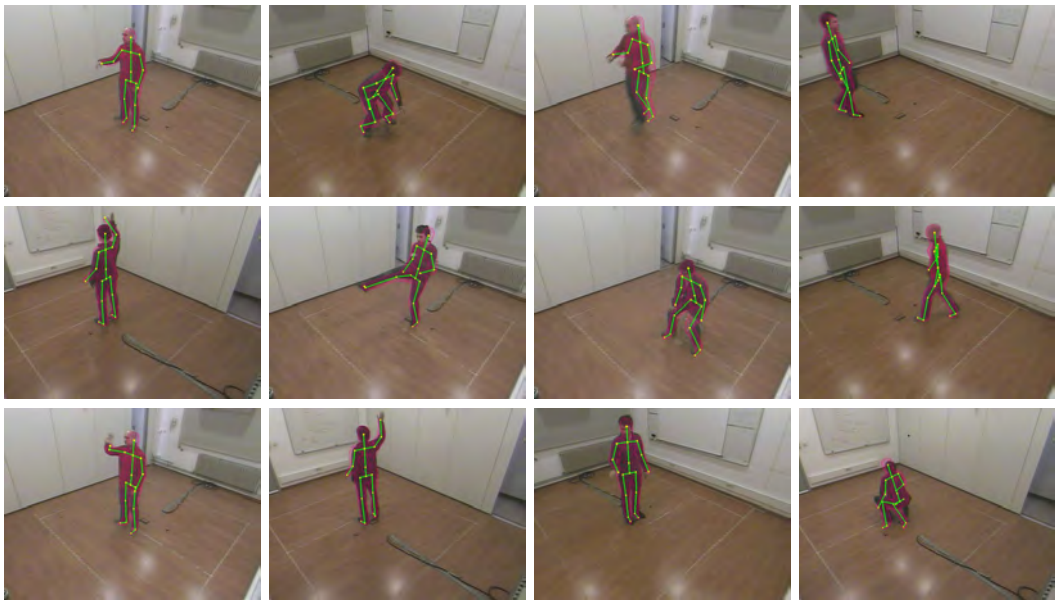
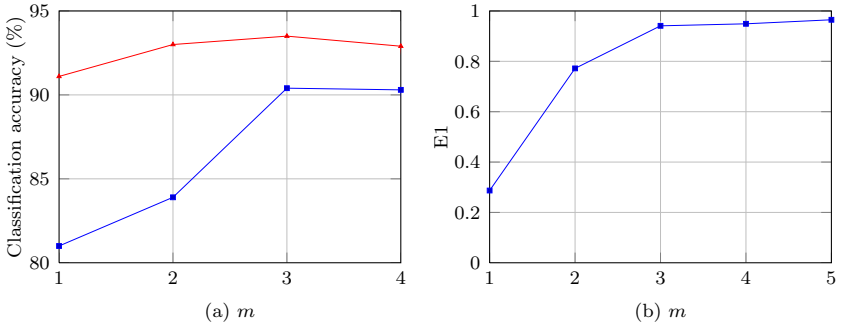


FIGURE 4.28: Selected tracking samples.



(a) m (b) m

FIGURE 4.29: Analysis of the overall recognition rate and the embedding dimensions in the Multi-View dataset (a) Recognition rate of the trajectory matching approaches as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the Multi-View dataset.

93.5% for $m = 3$ and the SVM approach. The proposed trajectory matching shows an excellent performance for action classification from markerless motion capture data. In contrast to the FL dataset experiments, we observe that the SVM classification consistently outperforms the weighted approach. A look at the confusion matrices (Fig. 4.30) reveals that our approach delivers an excellent accuracy with action categories involving large motions. Majority of misclassifications occur in action classes involving single arm motions, especially for *raise hand* and *wave hand*, and for class *jump*, which is mainly confused with some *crouch* and *sit* sequences due to the initial leg flexion before the jump starts.

In comparison to the SVM classification approach, the training method based on model variable fusion shows a lower performance with actions involving a single arm motion. We have observed that in these actions, right shoulder rotations get heavily weighted

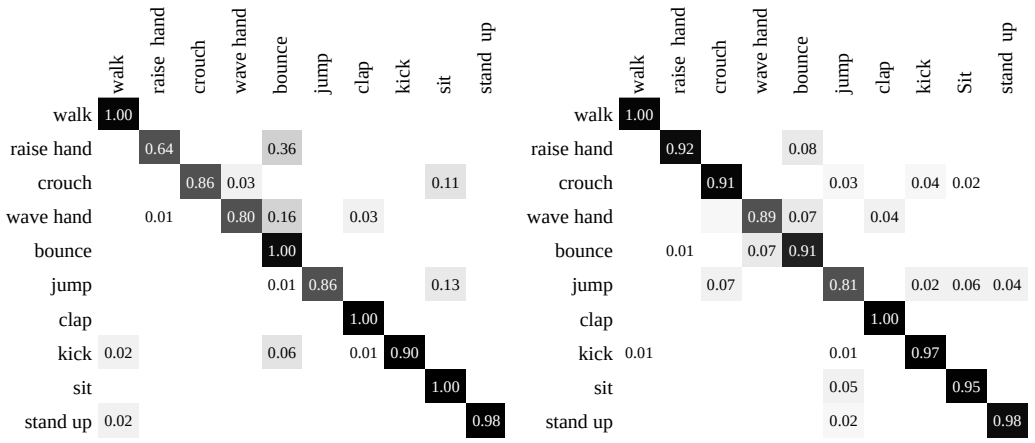


FIGURE 4.30: Confusion matrices in the Multi-View dataset for (a) trajectory matching and weighted approach (b) trajectory matching and SVM approach.

during training phase. Specifically, the actions *raise hand* and *wave hand* have high weights on x and y axes of rotation in the shoulder joint (in our model, the z rotation axis is defined along the upper arm), as it is a salient feature in the dataset. In the case of *wave hand* and *bounce*, right elbow joint is weighted over the mean, because the pattern in this joint is particular of these two actions. However, in action *bounce*, the shoulder's y-axis, a differential trait from *wave hand*, has no importance. This is actually because it is differential with respect to *wave hand*, but it is very similar to many other actions in the dataset. Similarly, elbow rotation in action *raise hand*, as a differential trait versus *wave hand* and *bounce*, is not significantly weighted during training for the same reason. To summarize, the weighted approach is able to focus on distinct human model variables in an automated manner, with effective results when large motion takes place in the performance of an action. However, weighting model variables in a discriminative manner considering them isolatedly has limitations with a single arm motion.

Provided that trajectory matching is rather exhaustive, and works on the basis of nearest neighbor search, we test the impact of the training set size, by varying the amount of training and testing data. Specifically, we test with training sets of 2/3, 1/2 and 1/3 of the available data, while the rest is left for testing. The average classification accuracy reported in this experiment is shown in Table 4.4. We highlight the notably high performance delivered by these approaches even when 1/3 of the data is used for training.

In a final experiment, we use the action sequences of 3 actors for training and the remaining for testing. In this way, we evaluate the performance of the algorithm under stylistic variations. In this last experiment, the average recognition rate is 84.3% with the SVM classification and 82.2% with the weighted approach, confirming that the

Training set size	SVM	Weighted
2/3	93.5	90.4
1/2	91.8	88.8
1/3	86.2	83.9

TABLE 4.4: Average classification accuracy for the Multi-View dataset using Trajectory Matching (embedding dimension $m = 3$ and parameter $\delta = 5$). The first column denotes the amount of training data used in the random folds.

dense matching of trajectories in the phase space is sensitive to some stylistic variations, although still providing a remarkable accuracy. In overall, the proposed method has a promising performance with multi-view video data in spite of some important tracking errors.

Using 2/3 of the data for training, our action recognition method takes an average of 0.3 s per frame in an Intel 2.4GHz CPU with 4GB of RAM (in test time, given the estimated poses).

Bag of Delay Vectors Following the evaluation procedure of the trajectory matching approach, we randomly split the data into training and testing sets containing 2/3 and 1/3 of the data respectively. We then apply the BoDV model to classify the test data. Average classification results are reported in Fig. 4.31. The best reported performance is 89.7%, and it is achieved with hierarchical clustering, $m = 3$ and $nw = 32$.

Interestingly, the k-means clustering delivers a surprisingly good classification accuracy for the single arm actions (*raise hand*, *wave hand*, *bounce*), with only a few confusions among these classes. However, this approach delivers a poor performance with *sit* and *stand* actions. Contrarily, the BoDV with hierarchical clustering behaves similarly to the trajectory matching approach, where single arm actions are confused more often.

The BoDV model delivers a lower classification accuracy than the trajectory matching approaches, even when using more training data. We hypothesize that the lower performance is mainly because BoDV performs a more simplistic analysis of the phase space: topology is completely overlooked. Additionally, the clustering methods employed in this dissertation might not be appropriate to handle delay-vectors obtained from noisy time series.

Random Forests in the Phase Space We follow the same evaluation protocol of the preceding experiments, consisting in generating random folds by splitting the data into 2/3 for training and 1/3 for testing. Differently from the mocap experiments of Section 4.6.1, we only employ joint angles for this experiment. Hence, the definition of the tests

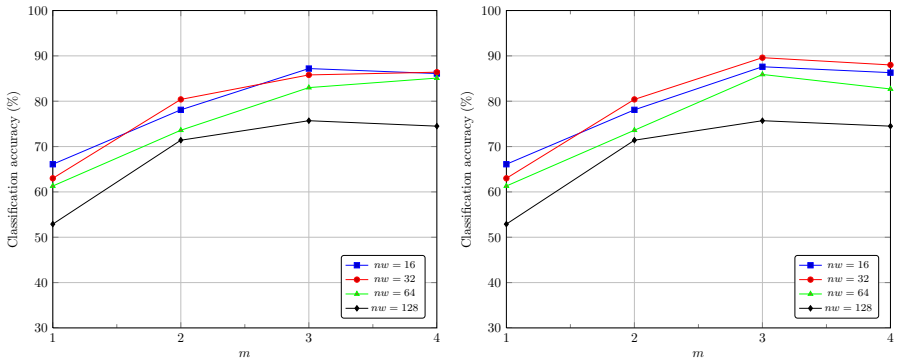


FIGURE 4.31: Average classification results in the Multi-View dataset. Accuracy is plotted as a function of the embedding dimension m (each curve represents a different number of words nw) (a) BoDV with k-means. (b) BoDV with hierarchical clustering

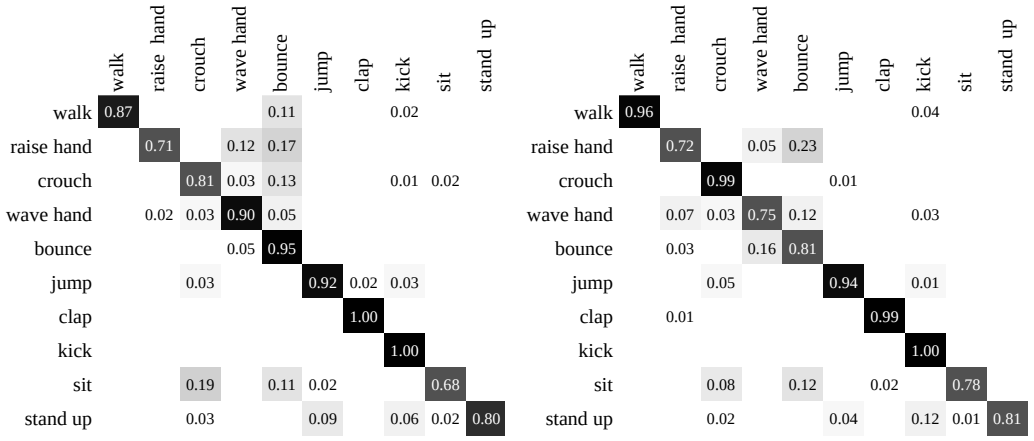
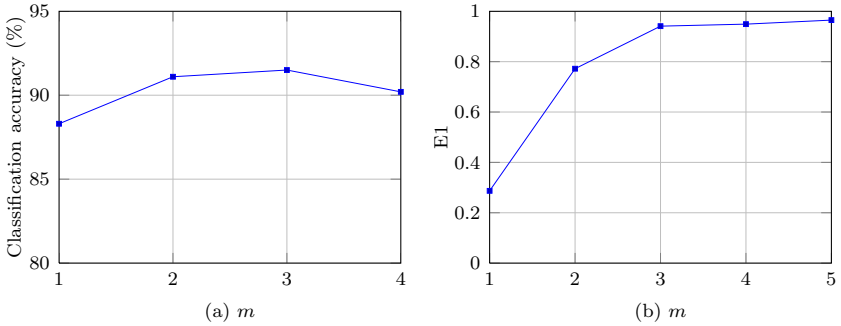


FIGURE 4.32: Confusion matrices in the Multi-View dataset for (a) BoDV with k-means (b) BoDV with hierarchical clustering. In both cases $m = 4$ and $nw = 32$

is similar to the one presented for the Weizmann dataset (see Section 4.6.2), but instead of using generic features, here the input space comprises the set of joint angles and the embedded space contains the delay-vectors obtained from multivariate time series. For evaluation, we follow the same protocol described in the previous experiments on this dataset. We sample patches having a length of 5 frames, and we train a forest of 10 trees with maximum depth 8. We perform 5000 random tests in each branching node (500 parameter sets and 10 thresholds per parameter set). Provided that we work with segmented action sequences, the classify a sequence with the action having more votes.

Results as a function of the embedding dimension are shown in Fig. 4.33. We report an average classification accuracy of 91.5% with the an embedding dimension of $m = 3$. This classification accuracy is slightly lower than the trajectory matching approach using SVM classification, but it is obtained much faster in test time. Similarly to the preceding experiments presented in this chapter, we observe a strong correlation between the performance and the E1 measure used to find the minimum common embedding



(a) m (b) m

FIGURE 4.33: Analysis of the overall recognition rate and the embedding dimensions in the Multi-View dataset (a) Recognition rate of the random forest approach as a function of the embedding dimension. (b) E1 measure [Cao97] to determine the Minimum Common Embedding dimension of the Multi-View dataset.

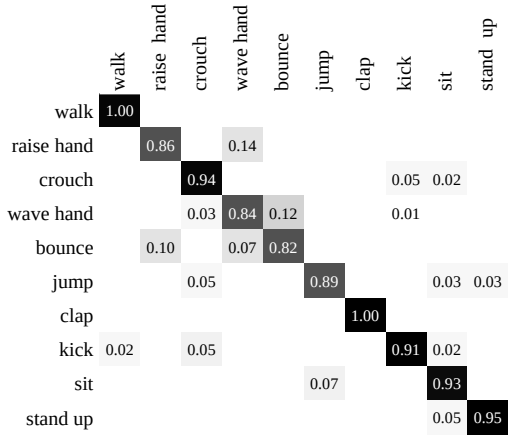


FIGURE 4.34: Confusion matrix in the Multi-View dataset for the random forest approach with $m = 3$

dimension. Furthermore, in this dataset, employing phase-space reconstructions yields around 3% gain in performance with respect to the input feature space.

A confusion matrix is shown in Fig. 4.34. As in the previous cases, most of the confusions are produced for arm actions involving a waving pattern, *bounce*, *wave hand*, *wave hand* and *raise hand* and also between *jump* and *sit*, since the initial phase of the jump has some similarities, in terms of pose, with the sitting motion.

4.7 Conclusions

We have presented a novel framework for action recognition based on time-delay embeddings. Although similar to a few existing approaches in the theoretical background, our approach differs in how the reconstructed space is interpreted and treated in order to model human actions. Our approach exploits both geometrical and topological structure of the embedded phase-space by means of a novel trajectory matching algorithm. On the other hand, we exploit the geometrical structure of the reconstructed

space by grouping delay-vectors in geometric loci. These loci can be obtained by clustering methods, and hence loci become a quantization of the reconstructed phase space. This quantization can be efficiently exploited using analogy of the popular bag-of-words model. Although efficient, several clustering methods fail to deliver an accurate and discriminative representation of the geometric loci of delay-vectors. For that matter, we propose a novel random forest approach that deals with reconstructed phase spaces of multivariate time series. Our experimental results show that these approaches achieve state-of-the-art performance on human action recognition problems involving articulated structures. Moreover, we outperform previous approaches using time-delay embeddings. We also conducted experiments with low-level features, obtaining a competitive performance. These results indicate that, although conceived for articulated models, the framework proposed in this chapter is suitable for generic time series mining.

5

Using Poses for Temporal Clustering of Human Actions

5.1 Introduction

In the previous chapter we have dealt with the problem of action recognition. In action recognition, there is usually a clear link between the perceived scene and the semantics, and that link is given by a single action label associated to an action example. However, understanding actions by analyzing motion in scenes is generally a more challenging problem, because a motion sequence needs to be segmented into behaviors.

The automated segmentation of a human motion sequence into plausible and semantically meaningful human behaviors is a central problem in computer vision and in computer graphics. Addressing this problem from the perspective of human poses obtained by motion capture systems is becoming more relevant due to the proliferation of motion capture databases and recent advances in markerless motion capture [GRBS10, BJB12, YGvG12, SFC+11]. Such an approach is not only interesting because of the potential availability of data, but also because human poses have potential for learning motion patterns that can be robustly employed across datasets and domains.

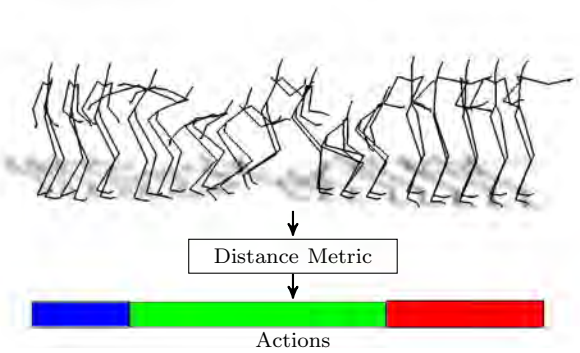


FIGURE 5.1: System Overview: Human motion sequences are clustered into different actions using a learned distance metric. We use annotations available in a mocap dataset to learn a distance metric that captures the semantic similarity between skeleton motion.

Segmenting human motion into distinct actions is a highly challenging problem. From the motion analysis perspective, segmentation is difficult due to large stylistic variations, temporal scaling, changes in physical appearance, irregularity in the periodicity of human motions and the huge number of actions and their combinations. From a semantic viewpoint, segmentation is inherently elusive and difficult because in the vast majority of cases it is not clear when a set of poses describes an action. For instance, punching with the left hand and punching with right hand can be different actions, but it might be also regarded as punching or even more general as boxing.

In this chapter, we propose to learn what makes a sequence of poses different from others such that it should be annotated as an action, as illustrated in Fig. 5.1. To this end, we make use of already annotated motion capture datasets and formulate action segmentation as a weakly supervised temporal clustering problem for an unknown number of clusters. Since publicly available datasets might contain different motions and action labels than the test sequences, we can not use the annotation directly for action segmentation. Instead, we use the annotations to learn a distance metric for skeleton motion using relative comparisons in the form of *samples of the same action are more similar than they are to a different action.* This is very intuitive since the sequences of a single database are usually labeled based on a semantic similarity. The learned distance metric is then used to cluster the test sequences. To this end, we employ a hierarchical Dirichlet process that also estimates the number of clusters.

The main advantage of our method is that it can be used for unseen actions and across datasets as we will show in our experiments.

5.2 Related Work

Metric learning from pose data has been mainly proposed in the computer graphics community in order to learn human-like perceptual similarities between poses [TLKS08]. The learned distance metric is then applied to the task of finding suitable transitions and content-based pose retrieval [CZXL09, CZN+11, WB03]. Metric learning has proven to be also useful in recognizing actions from video. Tran and Sorokin [TS08] extract silhouette and optical flow features from videos and they use them in conjunction with Large Margin Nearest Neighbors (LMNN) [WS09] to learn a metric that properly separates different action classes. More recently, Kliper-Gross et al. [KGHW11] have proposed a metric learning approach for one-shot learning of actions in videos.

In order to efficiently annotate actions in large collections of video or mocap data, some researchers have focused on unsupervised segmentation and clustering of human actions. Barbic et al. [BSP+04] propose a change detection algorithm for mocap data. They provide accurate results, but their method is not able to cluster the temporal segments into the different behaviors. Ozay et al. [OSC08] overcome the clustering problem by modeling the first three principal components of the data as an autoregressive model. The coefficients of the model are then clustered with k-means. Similarly, the Aligned Cluster Analysis (ACA) proposed by Zhou et al. [ZDH08] extends the k-means concept to cluster time series. They show that ACA can accurately find different behaviors in sequences of mocap data. However, [OSC08] [ZDH08] are limited by having to manually set the number of clusters (actions) k. In [RWS10], this limitation is tackled by using a spike-train driven dynamical model that can detect motion transitions and clusters them into different behaviors, without having to manually set the number of clusters k. As far as video data is concerned, approaches such as [KOSS11][KBvGF10] have proposed variants and extensions of hierarchical Dirichlet processes (HDP) [TJBB06] in order to find activities using optical flow features mainly. In [FSJW08], HDPs are used as a prior for HMM parameters in order to cluster time series data into distinct behaviors. This latter approach is applied to synthetic data, stock indices and dancing honeybee data.

5.3 Proposed Approach

We aim at a temporal clustering of human actions in which one can provide some knowledge learned from data. The training data might be from a different database containing actions that are not relevant for the testing data. To meet these requirements, we learn a distance metric from pose-based features, and we use this metric to cluster pose feature vectors (Section 5.3.2) as illustrated in Fig. 5.3a). The outcome of such a

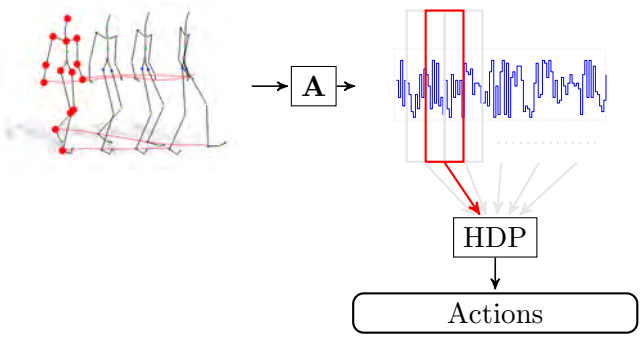


FIGURE 5.2: Detailed overview of our approach. A set of pose-based features are extracted using 14 relevant joints (marked with red spheres). These features are subsequently clustered into primitives using a metric ($\mathbf{A}$) learned on related action sequences. In order to infer the different actions in a sequence, we first group the primitives using a sliding window. Then, we provide the resulting sets of primitives to a hierarchical Dirichlet process.

clustering is then provided to a hierarchical Dirichlet process (HDP) in order to obtain the different activities of a motion capture sequence (Section 5.3.3). This strategy allows us to cluster motion sequences into different behaviors without knowing the exact number and types of actions in a test sequence.

5.3.1 Pose-based Features

The features employed in this paper are a rather simple yet efficient way of exploiting the pose information. We start by removing the orientation and translation of the input poses, in order to set them in a reference system that will allow an invariant comparison between action sequences. From these rotation and translation invariant poses, we obtain a set of 14 relevant joint positions $\{\mathbf{q}_1, \ldots, \mathbf{q}_{14}\}$ that can be easily obtained in different datasets [BSP+04]; see Fig. 5.2. These joint positions are used to compute the following feature vector:

$$\mathbf{x} = \{\mathbf{q}_1, \ldots, \mathbf{q}_{14}, \dot{\mathbf{q}}_1, \ldots, \dot{\mathbf{q}}_{14}, \ddot{\mathbf{q}}_1, \ldots, \ddot{\mathbf{q}}_{14}\} \tag{5.1}$$

where $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ denote joint velocity and acceleration respectively (derivatives are computed by time differences). In practice, we subsample mocap data (recorded at 120Hz) at 30 Hz.

5.3.2 Learning a metric for pose-based features

Given a set of feature vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in $\mathbb{R}^D$, we aim at learning a positive semi-definite matrix $\mathbf{A}$ such that the distance

$$d_A(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A} (\mathbf{x}_i - \mathbf{x}_j) \tag{5.2}$$

satisfies a set of constraints defined in terms of relative comparisons of the form "$\mathbf{x}_i$ is closer to $\mathbf{x}_j$ than to $\mathbf{x}_k$". Using action labels, we can formulate these constraints in terms of similarity and dissimilarity between triplets of feature vectors. Under such constraints, we learn the matrix $\mathbf{A}$ employing Information-Theoretic Metric Learning (ITML) [DKJ+07]. ITML finds a suitable matrix $\mathbf{A}$ by formulating the problem in terms of how similar is $\mathbf{A}$ to a given distance parameterized by $\mathbf{A}_0$ (typically, the identity or the sample covariance). Provided that (5.2) is a Mahalanobis distance, one can treat the problem as the similarity of two Gaussian distributions parameterized by $\mathbf{A}$ and $\mathbf{A}_0$ respectively. That leads to an information theoretic objective in terms of the Kullback-Leibler divergence between both Gaussians. This divergence can be expressed as a LogDet divergence [DKJ+07], thus yielding the following optimization problem:

$$\begin{aligned} \underset{\mathbf{A},\xi}{\text{minimize}} \quad & D_{ld}(\mathbf{A}, \mathbf{A}_0) + \lambda D_{ld}(\text{diag}(\xi), \text{diag}(\mathbf{c})) \\ \text{s. t.} \quad & \delta_{(i,j)}(\xi_{(i,j)} - \text{tr}(\mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T) \geq 0 \\ & \mathbf{A} \succeq 0, \xi \geq 0 \end{aligned} \tag{5.3}$$

where D_{ld} is the LogDet divergence, $\mathbf{c}$ is the vector of constraints, ξ is a vector of slack variables (initialized to $\mathbf{c}$ and constrained to be component-wise non-negative) that guarantees the existence of a solution and λ is a parameter controlling the tradeoff between satisfying the constraints and minimizing the similarity between distances.

In order to learn the metric (5.2) for the pose features (5.1), we have to define the constraints $d_A(\mathbf{x}_i, \mathbf{x}_j) \leq c_{(i,j)}$ or $d_A(\mathbf{x}_i, \mathbf{x}_j) \geq c_{(i,j)}$ for a pair of feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$. Since for each feature $\mathbf{x}_i$ we have only an action label y_i, we define the constraints based on triplets of points $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ with class labels (y_i, y_j, y_k), where feature vectors with the same label should be closer to each other than to feature vectors with different labels. Using $\delta_{(i,j)} \in \{-1, 1\}$ as similarity indicator (5.3), i.e., $d_A(\mathbf{x}_i, \mathbf{x}_j) \leq c_{(i,j)}$ if $\delta_{(i,j)} = 1$ and $d_A(\mathbf{x}_i, \mathbf{x}_j) \geq c_{(i,j)}$ otherwise, we formulate the following constraints:

$$\begin{aligned} y_i = y_j = y_k \quad & \delta_{(i,j)} = 1 \quad d_A(\mathbf{x}_i, \mathbf{x}_j) \leq \max(d(\mathbf{x}_i, \mathbf{x}_j), d(\mathbf{x}_i, \mathbf{x}_k), d(\mathbf{x}_j, \mathbf{x}_k)) \\ y_i = y_j \wedge y_j \neq y_k \quad & \delta_{(i,j)} = 1 \quad d_A(\mathbf{x}_i, \mathbf{x}_j) \leq \min(d(\mathbf{x}_i, \mathbf{x}_k), d(\mathbf{x}_j, \mathbf{x}_k)) \\ y_i \neq y_j \wedge y_i = y_k \quad & \delta_{(i,j)} = -1 \quad d_A(\mathbf{x}_i, \mathbf{x}_j) \geq d(\mathbf{x}_i, \mathbf{x}_k) \\ y_j \neq y_i \wedge y_j = y_k \quad & \delta_{(i,j)} = -1 \quad d_A(\mathbf{x}_i, \mathbf{x}_j) \geq d(\mathbf{x}_j, \mathbf{x}_k) \\ y_i \neq y_j \neq y_k \quad & \delta_{(i,j)} = -1 \quad d_A(\mathbf{x}_i, \mathbf{x}_j) \geq \min(d(\mathbf{x}_i, \mathbf{x}_j), d(\mathbf{x}_i, \mathbf{x}_k), d(\mathbf{x}_j, \mathbf{x}_k)) \end{aligned}$$

The values on the right hand side of the inequalities, $c_{(i,j)}$, are defined on the Euclidean distances $d(\ )$ between the features $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{x}_k$. When the features have the same or completely different labels, the distance is constrained to be less or greater than the Euclidean distances, respectively. When only two features have the same label, the

distance is constrained to be less than the Euclidean distances of the feature vector pairs with different labels.

For learning the metric, we randomly draw the triplets for generating the constraints from the training set. Furthermore, we estimate the tradeoff parameter λ by means of cross-validation, where our goal is to cluster pose-based features into a set of K primitives. To this end, we rely on a hierarchical clustering algorithm [War63] to overcome the dependency on the initial point. We set a sufficiently high K (typically ranging from 16 to 64 clusters) and we find λ by minimizing the *purity* of the clusters obtained in cross-validation:

$$\mathcal{C}(\lambda) = 1 - \frac{1}{K}\sum_{k=1}^{K}\frac{1}{n_k}\max_y(n_k^y) \tag{5.4}$$

where n_k is the number of feature vectors in the cluster k, and $\max_y(n_k^y)$ denotes the number of feature vectors of the class y appearing most frequently. Note that the dependence on λ comes from the fact that such parameter influences the resulting clusters.

Learning a metric from the proposed pose-based features can be seen as a *data-driven* transferring of implicit semantic distances derived from the class labels. In order to reduce the bias towards certain performance styles and to keep some temporal constraints, we investigate two additional variants of the pose-based metric learning framework.

Symmetry Unbiasing In order to reduce the bias towards action examples performed exclusively with right or left limbs, we *mirror* the poses. For instance, if we learn the metric with examples of *raising right hand*, we mirror the pose-based feature vectors in order to represent *raising left hand* and we assign the same action label (*raising hand*) to all these examples.

Temporal Alignment Two motion sequences of the same action class can be aligned by dynamic time warping [SC78]. Then, if under such alignment, a feature vector $\mathbf{x}_i$ from one sequence matches another feature vector $\mathbf{x}_j$ from the other sequence, we say that they are aligned. If two feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ belonging to the same action class are aligned, they should be more similar than a third feature vector $\mathbf{x}_k$ of the same class that is not aligned with $\mathbf{x}_i$ and $\mathbf{x}_j$. Therefore, for any randomly drawn triplet $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ such that $y_i = y_j = y_k$, we define the following inequalities:

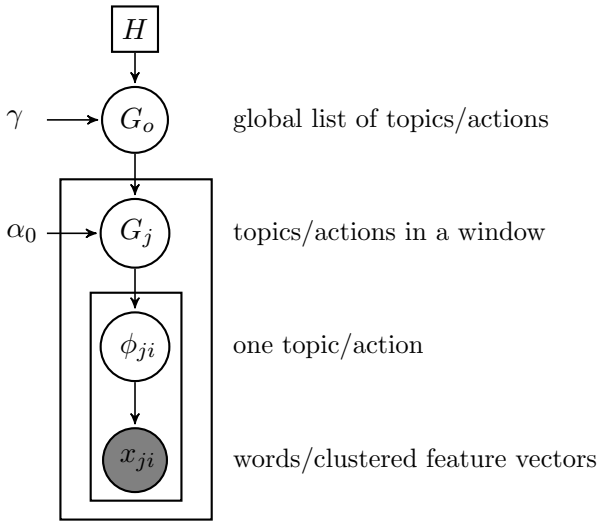


FIGURE 5.3: Detail of the hierarchical Dirichlet process.

$$
\begin{array}{rlll}
i,j,k & aligned & \delta_{(i,j)} = 1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \leq \max(d(\mathbf{x}_i,\mathbf{x}_j), d(\mathbf{x}_i,\mathbf{x}_k), d(\mathbf{x}_j,\mathbf{x}_k)) \\
i,j & aligned & \delta_{(i,j)} = 1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \leq \min(d(\mathbf{x}_i,\mathbf{x}_k), d(\mathbf{x}_j,\mathbf{x}_k)) \\
i,k & aligned & \delta_{(i,j)} = -1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \geq d(\mathbf{x}_i,\mathbf{x}_k) \\
j,k & aligned & \delta_{(i,j)} = -1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \geq d(\mathbf{x}_j,\mathbf{x}_k) \\
i,j & !aligned & \delta_{(i,j)} = -1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \geq \max(d(\mathbf{x}_i,\mathbf{x}_k), d(\mathbf{x}_j,\mathbf{x}_k)) \\
i,j,k & !aligned & \delta_{(i,j)} = -1 & d_A(\mathbf{x}_i,\mathbf{x}_j) \geq \min(d(\mathbf{x}_i,\mathbf{x}_j), d(\mathbf{x}_i,\mathbf{x}_k), d(\mathbf{x}_j,\mathbf{x}_k))
\end{array}
$$

where a pair of indices followed by *aligned* denotes a unique aligned pair within the triplet (!*aligned* expresses the contrary, the unique unaligned pair in the triplet) and three indices followed by *aligned* indicate that all the samples are aligned (a !*aligned* triplet means that any sample is aligned). These constraints replace the initial constraint for the case $y_i = y_j = y_k$, which was fulfilled from the beginning.

5.3.3 Discovering Actions

Given a sequence of pose-based feature vectors $\mathcal{X} = \{\mathbf{x}_0, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T\}$, we address the problem of inferring the performed actions as a temporal clustering problem in which we rely on weak supervision to learn semantic similarity in the form of a metric. Contrarily to other approaches [ZDH08], we want to address the temporal clustering problem for an unknown number of clusters or actions. For that matter, we rely on a hierarchical Dirichlet process (HDP) [TJBB06].

The HDP (see Fig. 5.3) is a nonparametric Bayesian model for clustering problems involving groups of data. We assume that we have J groups of data, each of which has n_j data points $(\mathbf{x}_{j1} \dots \mathbf{x}_{jn_j})$. Data points in each group are modeled with a mixture model and are assumed to be exchangeable. If we denote $(\Theta, \mathcal{B})$ a measurable space and G_0 a probability measure on that space, then the HDP is a distribution over random probability measures on $(\Theta, \mathcal{B})$. This set of measures comprises a probability measure G_j per data group and a global probability measure G_0. The global probability measure G_0 is distributed according to a Dirichlet process with base probability measure H and concentration parameter γ (also know as inverse variance). Each G_j is conditionally independent given G_0 with distribution $G_j \sim \mathrm{DP}(\alpha_0, G_0)$. The described model allows different groups to have different characteristics, represented by mixture models with different mixing proportions, HDPs also allow statistical strength by sharing mixture components among groups. Finally, each data point $\mathbf{x}_{ji}$ is associated with a factor ϕ_{ji}, with distributions given by $F(\phi_{ji})$ and G_j respectively. The complete model (Fig. 5.3) is given by:

$$\begin{aligned} G_0|\gamma, H &\sim \mathrm{DP}(\gamma, H) & G_j|\alpha_0, G_0 &\sim \mathrm{DP}(\alpha_0, G_0) \\ \phi_{ji}|G_j &\sim G_j & \mathbf{x}_{ji}|\phi_{ji} &\sim F(\phi_{ji}) \end{aligned} \tag{5.5}$$

In order to express the input feature vectors $\mathcal{X}$ as appropriate grouped data, we cluster $\mathcal{X}$ into primitives and then we employ sliding windows of a given length and overlap in order to obtain J groups of primitives from an input motion sequence. Hence, two clustering levels are considered. The *low-level clustering* aims at quantizing the feature vectors into K primitives, such that discrete data can be provided to the HDP. The *low-level clustering* is performed by combining a hierarchical clustering algorithm (see Section 5.3.2) with the learned metric $\mathbf{A}$. In contrast, the *high-level clustering* is the temporal clustering of the different actions. Using a topic modeling metaphor (see Fig. 5.3), *low-level clustering* is the step of computing *words* while the *high-level clustering* consists in finding the *topics* within a sequence. Actions are hence understood as co-occurring words in specific segments or groups (G_j) of the sequence (G_0). The implications of such a model are two-fold. First, we assume that the quantized feature vectors follow a multinomial probability distribution within each action and, consequently, temporal ordering is ignored. Second, the *low-level clustering* step is crucial, since producing better words will produce better clustering results.

In order to cluster the motion sequence into different actions, we employ an efficient Gibbs sampler for HDP mixture models [TJBB06]. This sampler is based on the Chinese

Restaurant Franchise (CRF) representation. Suppose n_j customers (corresponding to ϕ_{ji}) at a Chinese restaurant with an unbounded number of tables. The first customer sits at the first table, while the subsequent customers sit in an occupied table with a probability proportional to the number of customers already sitting around that table. Contrarily, they sit in the next unoccupied table with probability proportional to α_0. If a customer i sat at table t, we have the following conditional distribution:

$$t_{ji}|t_{j1},\ldots,t_{ji-1},\alpha_0 \sim \sum_t \frac{n_{jt}}{\sum_{t'} n_{jt'}+\alpha_0}\delta_t + \frac{\alpha_0}{\sum_{t'} n_{jt'}+\alpha_0}\delta_t^{new} \tag{5.6}$$

where n_{jt} is the number of customers sitting at table t. Once the customers have sat down, the result is a partition $\phi_{j1},\ldots,\phi_{jn_j}$. The generative process employed in order to draw the partition is known as the Chinese Restaurant Process (CRP) [TJBB06], and it is used to sample from a Dirichlet process. Now, in order to extend this process towards sampling a HDP, we associate a table t with a factor ψ_{jt} drawn from G_0. By assigning $\phi_{ij}=\psi_{jt_{ji}}$ independently for each group j we marginalize out G_j. Since factors ψ_{jt} are drawn from a $DP(\gamma,H)$, we apply again the CRP. Using similar nomenclature, now a customer associated with ψ_{jt} sits at a table k_{jt} according to the following distribution:

$$k_{jt}|k_{11},\ldots,k_{1n_1},k_{21},\ldots,k_{jt-1},\gamma \sim \sum_k \frac{m_k}{\sum_{k'} m_{k'}+\gamma}\delta_k + \frac{\gamma}{\sum_{k'} m_{k'}+\gamma}\delta_k^{new} \tag{5.7}$$

where m_k is the number of customers at the *top level restaurant*.

Finally, a table k_{jt} is associated with a draw θ_k from H yielding the assignment $\psi_{jt}=\theta_{k_{jt}}$. This completes the generative process for the ϕ_{ji}, where G_0 and G_j are integrated out.

To compute temporal segments, we employ a sliding window of a given length and overlap. Using validation sets of mocap data, we found that a window of 7-15 frames and 1/2 of overlap worked well. Similarly, we found that values for concentration parameters in the range of 0.5 to 1.0 for γ and between 1 to 2 for α_0 (see Fig. 5.3) provided good results. The base probability measure H (see Fig. 5.3) is a symmetric Dirichlet distribution of parameter 0.5 [TJBB06].

5.4 Experimental results

5.4.1 Experimental Setup

We conduct several experiments on two publicly available mocap datasets to show the effectiveness of our method. The first dataset is the CMU mocap dataset [Car]. This dataset contains a huge collection of motions performed by 144 subjects. Sequences include examples of one action as well as complex activities involving a combination of simple actions. One of the main drawbacks of this dataset is that the labeling of sequences is rather imprecise and the availability of action examples is biased towards locomotion mainly. The second dataset is the HDM05 dataset [HDM]. The HDM05 dataset contains more than three hours of motion capture data, involving more than 70 motion classes in 10 to 50 realizations executed by various actors.

In our experiments, we employ the following training sets:

CMU Sequences from several subjects containing examples of *walk*, *jump*, *run*, *boxing*, *drinking*, *lean forward to reach*, *bend* and *kicking a ball* actions. Examples of actions such as *boxing* and *jump* present a number of punching and jumping styles and variations.

HDM05 Sequences from the 4 available subjects containing examples of *walk*, *run*, *grab*, *kick*, *clap*, *jog*, *punch*, *hop* actions. These examples are taken from the cut sequences, and contain a huge variation of styles. For instance, action *clap* involves clapping in front of the torso and above head.

The testing sets are the following:

CMU Sequences 1 to 14 of subject 86 as in [ZDH08].

HDM05 We generate 10 long sequences by concatenating cut sequences not included in the training set.

5.4.2 Evaluation Metrics

Manually annotating different actions in a human motion sequence is a difficult task. Annotators have to precisely determine motion transitions and action labels. Without a specific guidance, the annotation variability for a dataset would make action labels useless. This also renders the evaluation a challenging task, since it is difficult to objectively determine the goodness of an approach given some labels with a potential annotation bias. We therefore employ several evaluation metrics to measure the accuracy of our approach.

	M1			M2			M3			Error k		
A	CMU	HDM05	**I**	CMU	HDM05	**I**	CMU	HDM05	**I**	CMU	HDM05	**I**
Normal	82.7	88.1	78.4	84.3	89.9	79.8	61.6	70.2	66.1	3.3	2.5	1.4
Mirror	88.0	90.3	78.4	89.7	92.2	79.8	67.4	70.5	66.1	2.9	2.8	1.4
Mirror+Align	86.9	89.5	78.4	88.5	91.3	79.8	67.2	69.4	66.1	3	2.8	1.4

TABLE 5.1: Clustering results for the HDM05 concatenated sequences. For each one of the proposed metrics, we show the results when learning a metric on the CMU and HDM05 datasets and when using the Euclidean distance (**I**). See Section 5.4.2 for the definition of the evaluation metrics.

	M1			M2			M3			Error k		
A	CMU	HDM05	**I**	CMU	HDM05	**I**	CMU	HDM05	**I**	CMU	HDM05	**I**
Normal	87.3	89.5	88.5	88.2	90.5	89.4	73.9	82.2	80.6	3.1	2.5	2.5
Mirror	88.8	90.9	88.5	90.0	91.9	89.4	77.0	80.4	80.6	2.8	3.1	2.5
Mirror+Align	89.5	90.5	88.5	90.5	91.5	89.4	78.2	81.2	80.6	3.1	3.1	2.5

TABLE 5.2: Clustering results for the CMU sequences (14 sequences of subject 86). For each one of the proposed metrics, we show the results when learning a metric on the CMU and HDM05 datasets and when using the Euclidean distance (**I**). See Section 5.4.2 for the definition of the evaluation metrics.

Firstly, we use the same metric as [RWS10], that does not penalize oversegmentation as far as the estimated labels consistently match different actions. Since in [RWS10] the transitions are not evaluated, we use two versions of this evaluation metric. The first version (M1) evaluates all the frames, whereas the second version (M2) does not take into account the frames around ground truth transitions (we simply remove 0.2 seconds around the transition). The third evaluation metric (M3) is that of [ZDH08] applied to the case where the number of found clusters may differ from the ground truth. We compute the best label assignments for the number of clusters provided by the ground truth, hence under- and oversegmentation are strongly penalized. Finally, we provide the average error in the estimated number of clusters (Error k).

5.4.3 Experiments and Discussion

We learn different metrics employing the two training sets described in the previous section. Specifically, we learn metrics on the CMU and HDM05 training sets and we cross-test each of them on both the HDM05 and CMU test sets. Note that in learning a metric with CMU data, we use less labels than in the CMU test data (actions such as *stretching*, *basketball dribble* or *climbing a ladder* are not present in the training examples). Additionally, we investigate the impact of mirroring and alignment. In all the experiments, we employ a sliding window of 15 frames with 1/2 overlap and 21 primitives or words. We test with two sets of HDP concentration parameters, $\gamma = 0.7, \alpha_0 = 1$ and $\gamma = 1, \alpha_0 = 2$. We provide the average performance over these two sets of parameters. Results are shown in Tables 5.1 and 5.2.

Method	Known k?	Accuracy
ACA [ZDH08]	Yes	92.1% [1]
SAR [OSC08]	No	72.3% [2]
STS [RWS10]	No	90.9% [2]
Our HDP-E	No	89.4%
Our HDP-$\mathbf{A}_{\text{CMU}}$	No	90.5%
Our HDP-$\mathbf{A}_{\text{HDM05}}$	No	91.9%

TABLE 5.3: Comparison to state-of-the-art approaches on the CMU dataset. HDP-E stands for hierarchical Dirichlet process using Euclidean distance for feature-vector clustering while HDP-$\mathbf{A}_{\text{Z}}$ means that feature-vector clustering is performed with the metric learned with Z data. Note that methods are not directly comparable since they rely on different assumptions.

The performance on the HDM05 cut sequences (Table 5.1) shows that using a metric to cluster the feature vectors boosts the performance of the HDP temporal clustering. Best performance is achieved when using a metric learned on the HDM05 dataset. Such an outcome was expected, since action labels are the same as in the test data. Interestingly, using a metric learned with CMU data outperforms the Euclidean distance on the HDM05 test sequences. In both cases, we observe that, although the rest of the metrics show a superior performance, the error in the estimated number of clusters is higher when using a learned metric. However, the clusters provided by using the Euclidean distance also imply a higher number of mismatches between cluster labels and ground truth labels. When using the Euclidean clustering, actions such as *walk* and *jog* often get merged together into the same cluster. Such errors cause the number of estimated clusters to be closer to the ground truth, but several of the obtained clusters are lacking semantic meaning, as rather different labels get merged. On the contrary, although over-segmenting some actions into different stylistic performances, using the learned metric generally provides semantically meaningful clustering of motion into different behaviors.

Results on the CMU sequences of subject 86 confirm that using a metric provides better temporal clustering results. Interestingly, the best performance is achieved by learning a metric on the HDM05 dataset (see Table 5.2). This result yields two conclusions. First, the learned metric provides a good performance across datasets. Second, the benefits of learning a metric for temporal clustering of actions not only depend on the extent to which the training data could potentially explain the test data, but also on the labeling precision of the training examples.

Clustering results for the CMU test sequences are provided in Fig. 5.4. Using a learned distance metric for clustering the pose-based feature vectors involves a more semantically meaningful clustering of motion into actions. This can be clearly observed in sequences 1

[1] Computed using the software provided by [ZDH11]
[2] As reported in [RWS10]

to 9 and 11, where a number of noisy transitions are clustered as distinct behaviors when using the Euclidean distance. The exception to this performance is found in sequences 12 and 13. In these sequences, the metric learned from the HDM05 dataset helps in clustering action walk (red label in sequence 12 and 13 of Fig. 5.4) from the rest of the actions, but the examples employed in learning the metric do not help in achieving a semantically correct clustering of classes such as sweeping and dragging, and hence transitions between such actions, or even phases of the same action, are clustered as different behaviors.

When comparing the performance using mirroring and alignment constraints in Tables 5.1 and 5.2, we see that mirroring the poses improves the performance. The alignment improves the results only for training and testing on CMU; otherwise the performance degrades. This indicates that the alignment is only beneficial when the training sequences are not precisely segmented and labeled as it is the case for the CMU sequences.

In Table 5.3 we provide a comparison between state-of-the-art approaches for temporal clustering of human actions. In this comparison, we report the results using metric M2, since is the most similar to that employed to evaluate [OSC08] and [RWS10]. Note that the results provided by [RWS10] are computed on a subset of sequences (1-3 and 5-6) for subject 86, which are easier to segment than the other sequences (see Fig. 5.4). In spite of that, we report a better overall performance. We also show that our approach is a compelling alternative to ACA, since we can obtain accurate clustering results by resorting to action examples from other datasets instead of requiring the exact number of clusters.

5.5 Conclusions

In this chapter, we have presented an approach for temporal clustering of human behaviors. The method is based on learning a metric from pose-based features, such that the semantics of action labeling are learned in the form of a distance. Our experimental results have shown that the learned metrics improve the clustering results even across datasets and do not require that the actions of the test sequences are present in the training data. The benefit of the learned metric, however, depends on the similarity of the poses in the training and test set, but also on the labeling precision of the training examples. While this needs to be addressed in the future, the proposed approach, which exploits publicly available mocap datasets for temporal clustering, is a compelling alternative to unsupervised methods.

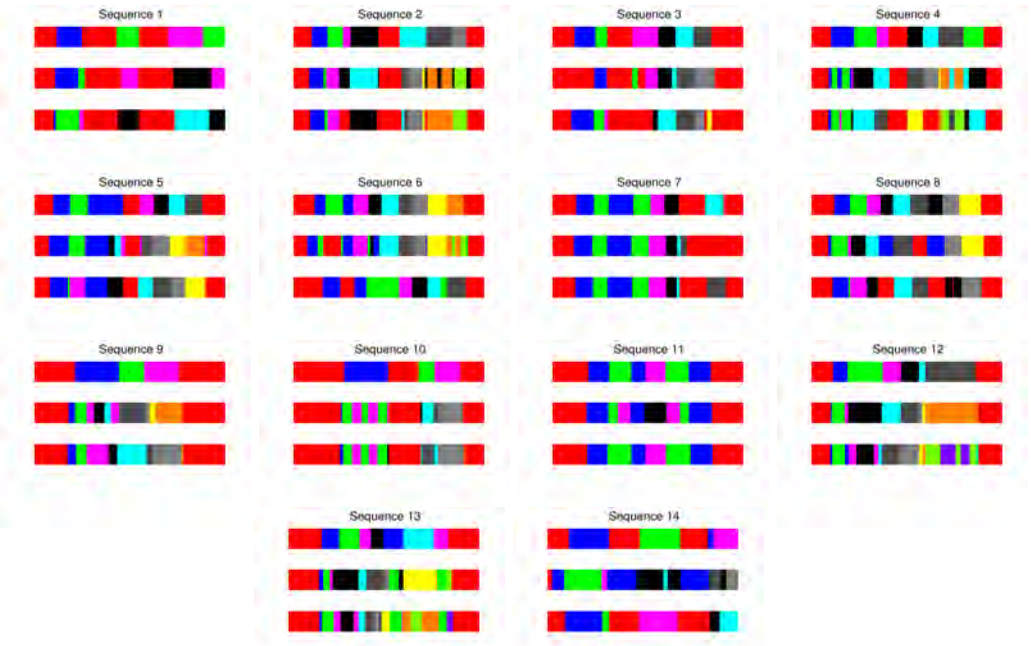


FIGURE 5.4: Temporal clustering of the 14 subject's 86 CMU sequences (best viewed in color). In each caption (top row) Ground Truth Labels (obtained from [ZDH11]), (mid row) temporal clustering with HDP and (bottom row) temporal clustering with HDP + metric learned with HDM05 data. The mocap sequences can be viewed at http://mocap.cs.cmu.edu/search.php?subjectnumber=86.

6

Conclusions

We conclude this dissertation on the use of articulated models for human motion analysis and behavior understanding by drawing the contributions of our work. The objective in this thesis has been to analyze a methodology employed in the analysis of human motion with a long tradition in science, arts or biomechanics, but with somewhat lack of popularity in computer vision in recent years. Our premise was that the rapid progress and cost reduction of capture devices, along with the massive availability of data, would raise again interest on this kind of analysis based on articulated models. For that matter, we contributed to this analysis with several algorithms for pose inference, action recognition and motion segmentation based on articulated models.

6.1 Contributions

1. **3D Pose estimation** : Estimating the human pose is a crucial step for employing articulated models for motion and behavior analysis. In this thesis, we focused on analysis-by-synthesis approaches for 3D pose estimation from multiple views or range data. This means that an articulated model is also necessary to generate hypothesis that will be matched with image observations. In this thesis, we use articulated models to reflect the prior knowledge about the underlying body structure and its motion, but also to formulate the pose estimation problem in terms of

a hierarchical relation between pose variables. Furthermore, used in conjunction with shape models, we can avoid extensive learning of discriminative features in order to locate every part of the body. Although this approach has shown to be feasible for depth data [SFC+11], it is still a huge learning problem for color images and more specifically for color images in multiple views. Our contributions to all the above-mentioned problems and challenges are summarized as follows:

- We propose a layered particle filtering approach that generalizes existing approaches of the state-of-the-art. This layered framework exploits the implicit hierarchy of articulated structures in order to perform efficient pose inference. Our generalization allows to perform hierarchical filtering with multiple passes on several state-space variables, which is beneficial for pose variables, but also for anthropometric variables. This formulation can be easily brought to an implementation allowing for flexible configurations of layers. In a pose estimation context, our layered particle filtering can be seen as a layered stochastic optimization method. For that purpose, we propose combinations of stochastic optimization with local optimization methods for pose refinement.
- Although detecting body parts from multiple views is a hard problem in general, we can still benefit from body part detectors, even if only a few parts can be detected. For that matter, we proposed a detector-driven layered particle filter providing state-of-the-art performance without learning motion models. We contribute with a method to robustly fuse detections in multiple views in order to obtain 3D body part detections. In our detector-driven algorithm, the articulated model is used in conjunction with inverse kinematics in order to draw poses from end-effectors.
- We proposed an approximate partitioning of the observations, which are images in multiple views or range images. Such partitioning helps in improving the observation model and provides a more accurate and faster pose inference.

Publications: The following publications in international conferences are related to the abovementioned topics [LMAPC11a, LMAPC11b, ALMCP11, ALMPC11, NLMAC12]. During this thesis, we have also produced a number of technical reports related to motion capture for European project ACTIBIO [FP7].

2. **Action Recognition** : This problem is posed as inferring the label of a human motion sequence given some labeled training examples. In this thesis, we address such problem by means of articulated models. To this end, we have first analyzed the view-invariance properties of such a representation, and we analyze sets of variables that can be used for motion analysis. Then, we have proposed the use of time-delay embeddings towards the characterization of action sequences, since

human motion can be seen as set of multivariate time series. Our contributions to these topic are the following:

- A novel approach towards exploiting time-delay embeddings. Time-delay embeddings are a principled method to reconstruct dynamical systems from data. Our approach consists in taking advantage of the geometry and topology of these systems in order to perform action recognition. or this reason, we propose reconstructing several dynamical systems in the same space in order to be compared. Compared to [ABS07], we do not rely on chaotic invariants, which are sensitive to noise and short-time observations.
- A distance between reconstructed phase spaces, that takes ideas from dynamic programming in order to find a topologically suitable alignment between phase-space trajectories. We use this distance with two classification strategies in order to recognize actions from human motion sequences.
- An action classification method based on constructing codebooks with delay vectors. The classification approach is inspired in the popular bag-of-words model, but in this thesis words implicitly encode dynamical information.
- A random forest approach that exploits joint mulivariate time-delay embeddings. The proposed algorithm is accurate and efficient, and delivers state-of-the-art performance in several action recognition datasets.

Publications: The result of this work has been published in [LMC12b].

3. **Temporal Clustering of Human Motion** : The huge amount of available human motion sequences and video action sequences makes sometimes unfeasible to apply classification approaches to infer actions. This infeasibility is due to the existence of possibly unseen or new action categories or rather different styles and variations in the performance of the same action. Therefore, the challenge is to obtain a set of behaviors with weak supervision. For that matter, we formulated the problem as a weakly supervised temporal clustering of human motion. Under such a formulation, we would like a system to be able to learn *what makes an action different from another* in a rather semantic manner. Our ultimate goal is that such information can be used even if the exact labels employed in learning are not present in the test data. In addressing this problem, we have made the following contributions:

   - An approach to learn *what makes some actions different from others* by means of metric learning. The novelty of this approach is that the ultimate goal is to apply the learned metric to unseen data, containing possibly unseen action categories. Hence, we expect the metric to gather some semantics in order to

quantize unseen motion sequences into meaningful motion primitives. To this end, we propose a set of constraints specifically suited for articulated models.

- A framework that combines metric learning and hierarchical Dirichlet processes in order to cluster motion sequences into actions or behaviors. Our approach has the advantage that can be used across datasets without providing the number of clusters or actions. Our experiments show that we can successfully clustering motion into actions across datasets. We have also shown that our approach is a compelling alternative to existing unsupervised methods [RWS10, ZDH08, ZDH11].

Publications: The result of this work has been published in [LMCvGL12].

6.1.1 Side Contributions

Apart from the abovementioned work, during the period of elaboration of this thesis, we have contributed to other fields with several algorithms and systems that have been reported in papers and reports:

- **Virtual view appearance representation for human motion analysis in multi-view environments**: In order to overcome the view dependency problem in action recognition from multiple-views, we proposed a novel virtual view generation relying on visual hull reconstructions. Although being only an approximate representation of the human shape, such a method allows having a set of constant perspectives of an action. In contrast to training from the original views, this new set of views allows a reduction of the number of training examples, since actions are seen from the same perspectives. This work was accepted in an international conference [LMCFC10].

- **Gesture recognition from depth data** : The recent arrival of consumer depth cameras has attracted the interest of the computer vision community towards developing algorithms and methods for scene understanding from depth data. This data has the intrinsic advantage of easily coping with the chronic problem of normal imagery: illumination. Due to these reasons, and in the context of European Project Fascinate [Eur], we have developed gesture recognition technologies. On the one hand, we targeted still gestures, as they have an enormous potential for interaction (they are easy to perform by users, require less physical effort) and we can formulate the problem in terms of object recognition. We advocate the use of class-specific random forests in combination with an efficient boosting approach in order to robustly locate such gestures in range images. Our approach

does not require segmentation of the input data and it works in real-time. Our results, both with offline and online data, show that the proposed approach is very robust to false positives while providing a fluid gesture-based interaction. When deployed on a live demo, our system runs in real-time for 4-5 gestures without using GPU implementations of random forests (which might yield super real-time performance). On the other hand, we have collaborated in the development of a real-time hand gesture and fingertip localization system. The underlying idea is to bring the successful trackpad paradigm to touchless interaction. This hand-based system has been also deployed in several live demos, and has been applied to different applications, including gaming and educational applications. The result of this work has been published in [LMC12a] and [SALM+ss].

6.2 Future Work

Although the methods presented in this thesis have provided good results in real problems of vision, there are still some lines of research that can be followed taking our work as a starting or reference point.

We have formulated pose estimation as a tracking problem. This means that an initial pose is needed. Furthermore, appropriate anthropometric proportions are required in order to have a good likelihood approximation. Despite providing a data-driven framework for tracking, that could potentially be used for the initialization of a body model, robust initialization is an open problem in many situations. In spite of that, our layered filtering framework has been used for the estimation of good initial poses and anthropometric profiles given a coarse estimation of the initial values for these unknowns. This research line has been addressed more in depth by Marcel Alcoverro in [ACP10] and in his thesis.

The main issue in 3D pose inference is that we have presented algorithms for multiple views or range data, i.e, situations where 3D can be estimated thanks to the technology behind the sensor or the sensor network. However, 3D estimation from single images or monocular videos is still an open problem. Therefore, the development of robust 3D pose estimation for single images and moving cameras is an exciting problem that can be tackled also using some articulated modeling framework. In order to tackle this problem, the use of body part detectors, in a similar way as proposed in this thesis, seems a promising line. However, the main challenge would be to map 2D information to the 3D world coordinate system from this kind of imagery. For moving cameras, Structure-from-Motion methods [DSTT00] provide a solid basis for 3D estimation, hence the framework proposed in this thesis can find application to moving cameras.

Regarding action recognition, we have investigated methods solely relying on articulated model features. However, in many cases actions are strongly conditioned and defined by the context. Context comprises the physical scenario, objects, other people and also the temporal ordering of actions. Research on methods and features exploiting context is a promising and interesting line. During the development of this thesis, we briefly investigated features that related an articulated model with the context. This research is still not finished, but some preliminary results pointed out that context information can help in improving the results of a baseline action recognition system.

As far as temporal clustering of human actions is concerned, our results show that metrics help in understanding actions from articulated model features. Furthermore, these results show that metrics learned in a dataset can be straightforwardly used in a new dataset. However, we have also seen that the actions in the two datasets must be related to some extent, and that metrics must be learned on datasets having accurate and precise labeling of actions. We have also seen limitations of the ITML framework in learning metrics in really high-dimensional feature spaces. Due to all these reasons, we think that research on low-rank metric learning approaches for human body model features could provide even more powerful results to the weakly supervised temporal clustering framework proposed in this thesis.

Finally, considering the strong relation between weakly supervised action clustering and action recognition, we consider that research on a joint probabilistic model for both action recognition (for known action labels) and segmentation of temporal clustering (in known actions and different unknown actions) would be useful for finding behaviors in motion sequences. Although hierarchical Dirichlet processes are really good for finding behaviors, they are based on assumptions that may not apply for all the actions (for instance, temporal order is overlooked). Additionally, our belief that a joint model would be useful is motivated by the fact that annotating actions is a elusive task, often dependent on the context. Therefore, it becomes extremely challenging for a machine to perform this task without some basic guidance. However, once this basic guidance is provided, an algorithm should not be only capable of clustering different behaviors but also annotating them according to known labels.

A

Twists and Exponential Maps

In this appendix, we review some fundamental mathematical concepts involving articulated models. We start by overviewing the characteristics of a kinematic chain structure to then detail the mathematical framework employed in order to generate human poses by means of articulated models.

A.1 Kinematic Chain Framework

The design of a model-based motion capture system implies implementing explicitly the prior knowledge about the body pose configurations. The body modelling framework must accomplish several requirements:

- **Provide a simple and compact representation of the possible body poses.** The relationship between body part locations and the body parameters must be simple and unique, while keeping a relatively low number of body parameters.
- **Capability of incorporating motion constraints**. Constraints can be set at the tracking level, but it is desirable that the body model incorporates the majority of them, thus leading to a more efficient tracking scheme.

The kinematic chain framework satisfies both requirements. Every 3D part location can be easily determined by the product of the twists affecting the motion of that point and most of the problem constraints are incorporated by means of hard kinematic restrictions.

In the following, we present twists and exponential map formula for kinematic chain framework and we introduce its application in an articulated human body model.

A.2 Twists and the Exponential Map Formulation

A kinematic chain [Par94] is an assembly of joints with several degrees of freedom (DOF), connecting rigid segments. Let SE(3) be the Euclidean group of rigid body motions and SO(3) the group of 3x3 proper rotation matrices. Both groups are Lie groups thus presenting an associated Lie algebra [Hal03].

Let us consider the rotation of one segment with respect a single DOF of a joint to which the segment is connected. To this end, we model this DOF as a rotation axis and we consider two elements on it: a unit vector $\omega \in \mathbb{R}^3$ and a point $\mathbf{q} \in \mathbb{R}^3$. Assuming unitary velocity of rotation, the velocity $\dot{\mathbf{p}}$ of a point $\mathbf{p}$ on a rigid object about the rotation axis is determined by:

$$\dot{\mathbf{p}} = \omega \times (\mathbf{p} - \mathbf{q}) \tag{A.1}$$

If we re-write the above expression in homogeneous coordinates we obtain the SE(3) Lie algebra $\hat{\xi}$:

$$\begin{aligned} \begin{bmatrix} \dot{\mathbf{p}} \\ 0 \end{bmatrix} &= \begin{bmatrix} \hat{\omega} & -\omega \times \mathbf{q} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \hat{\xi} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \\ \underline{\dot{\mathbf{p}}} &= \hat{\xi}\underline{\mathbf{p}} \end{aligned} \tag{A.2}$$

where $\underline{\mathbf{p}} = \begin{bmatrix} \mathbf{p} & 1 \end{bmatrix}^T$ is the point $\mathbf{p}$ in homogeneous coordinates and $\hat{\omega}$ is the Lie algebra of SO(3):

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{A.3}$$

Hence, Lie algebra of SE(3) is representing a twist. The solution for the differential equation in homogeneous coordinates A.2 leads to the connection between the Lie algebra and the Lie group:

$$\underline{\mathbf{p}}(\theta) = e^{\hat{\xi}\theta}\underline{\mathbf{p}}(0) \tag{A.4}$$

Any element $\hat{\xi}$ of the Lie algebra can be mapped to its corresponding rigid body motion by means of an exponential. Hence, the above solution shows the transformation from an initial location of the point to the current location after rotating θ radians, by means of the exponential map associated to the twist $\hat{\xi}$.

For an open kinematic chain with n axes of rotation, this formulation provides an interesting property. Let $\underline{\theta} = [\theta_1 \dots \theta_n]$:

$$g_P(\underline{\theta}) = e^{\hat{\xi}_1\theta_1}....e^{\hat{\xi}_n\theta_n}g_P(\mathbf{0}) \tag{A.5}$$

where $g_P(\theta)$ is the transformation from the rotations θ to the 3D locations of the chain points. This property allows us to compute the 3D location of every point in the chain by means of a product of the exponential maps associated to previous joints in the chain and a reference configuration $g_P(\mathbf{0})$. Moreover, this product is independent of the order in which it is computed.

An interesting property of Lie groups is that the exponential of its matricial elements is the matrix exponential. Therefore, we can compute the elements of an exponential mapping matrix by means of Taylor expansions, i.e, $\exp(\hat{\xi}) = \mathbf{I} + \hat{\xi} + \frac{\hat{\xi}^2}{2!} + \dots$. As a consequence we can develop the terms of the mapping as follows:

- Let $\|\hat{\omega}\| = 1$ where $\|\cdot\|$ is the Euclidean norm. Then, for any $\theta \in \mathbb{R}$

$$\mathbf{R} = e^{\hat{\omega}\theta} = \mathbf{I} + \sin\theta\hat{\omega} + (1 - \cos\theta)\hat{\omega}^2 \tag{A.6}$$

- Let $\|\hat{\omega}\| = 1$ and $\mathbf{v} = -\omega \times \mathbf{q}$. Then, for any $\theta \in \mathbb{R}$

$$\exp\left(\begin{bmatrix}\hat{\omega} & \mathbf{v}\\ 0 & 0\end{bmatrix}\theta\right) = \begin{bmatrix}\mathbf{R} & \mathbf{t}\\ 0 & 1\end{bmatrix} \tag{A.7}$$

  where

$$\mathbf{t} = \left(\theta\mathbf{I} + (1 - \cos\theta)\hat{\omega} + (\theta - \sin\theta)\hat{\omega}^2\right)\mathbf{v} \tag{A.8}$$

Note that if the rotation axes ω are aligned with the world coordinate system, the rotation matrices are very easy to find, because they will correspond to an x, y or z rotation matrix. Since this choice verifies $\hat{\omega}^2 = \mathbf{I} - \omega\omega^T$ and $\hat{\omega}\hat{\omega}^2 = -\hat{\omega}$, we obtain the following result:

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} e^{\hat{\omega}\theta} & -(1-\cos\theta)\hat{\omega}^2\mathbf{q} - \sin\theta\hat{\omega}\mathbf{q} \\ 0 & 1 \end{bmatrix} \tag{A.9}$$

where $\mathbf{M} \in \mathrm{SE}(3)$ is used to denote the exponential map.

B

Real-Time Upper Body Tracking with Online Initialization using Layered Particle Filters on Range Data

In this appendix, we detail how we brought our layered particle filtering framework to a real-time upper body tracking system. In order to achieve real-time performance, we propose a GPU iplementation of the evaluation step of our LPF. This evaluation step takes advantage of the high paralellization of a particle filter and the efficiency of our approximated partitioning of observations approach.

B.1 GPU implementation

Based on existing GPU implementations for PF algorithms aiming at articulated tracking [CCPM09], we propose an implementation of the APO-HPF approach presented in Chapter 3 for human body tracking with range data. Specifically, we opt for implementing the likelihood evaluation using OpenCL [TNI+09] with OpenGL interoperability. The reason for such a combination is to take advantage of a general purpose programming language as OpenCL and the rendering capabilities of OpenGL.

We start at the first layer or partition of the HPF with the torso particles. In this layer, we compute a squared 2D bounding box containing the upper-body. This bounding box is computed by approximating the projection of a 3D box that encloses the upper-body. However, we constrain the projected bounding box to be of a specific size, since the GPU implementation benefits from using bounding boxes having sizes being powers of 2. Our first approximate partitioning of the image covers approximately 1/4 of the image resolution and it is centered at the projection of the estimated body model centroid in the previous frame. For each particle, we render the model directly to a depth texture and then we use Quad primitives to map the pixels enclosed by the 2D bounding box to a bigger RGBA texture that we call the *mosaic texture*. The objective of this latter texture is to gather all the particle *tiles* that must be evaluated in one layer (see Fig. B.2). After mapping a particle onto its corresponding *tile* on the *mosaic texture*, we load the bounding box offset onto GPU global memory. We repeat this procedure until the *mosaic texture* is filled, yielding the maximum number of particles per layer.

We efficiently share the *mosaic texture* with OpenCL through its interoperability mechanisms, and we compute the pixel-wise depth and XOR costs (see Chapter 3 Sections 3.6.1 and 3.5.2). The OpenCL implementation uses different threads to perform these two costs, so that pixels are processed in parallel. Specifically, each thread looks for the corresponding pixel in the input depth map using its position in the *mosaic texture* and the offset previously loaded onto global memory. In this way, we compute both depth and XOR costs with one single read of the texture. The results of both costs and the evaluated pixels are stored in global memory (see Fig. B.2). After obtaining the pixel-wise differences, we perform a modified 2D sum-reduction on OpenCL. This version of the well-known sum-reduction is constrained to provide the cost of every particle instead of the sum of all the pixel values.

The described method is repeated in the remaining arm layers, with appropriate bounding boxes (see Fig. B.1).

B.2 Detector-Driven Filtering

We apply the detector-driven layered particle filtering approach presented in Chapter 3 Section 3.5.2 in order to incorporate body part detections in order to generate better particles. Our targeted body parts are, in this case, hands and head.

In order to detect the upper-body end-effectors we employ a filtering strategy based on [ALMPC11]. The key idea is to retrieve a set of salient geodesic extrema of the depth map on a per frame basis. This method starts by defining a graph $G = (V, E)$

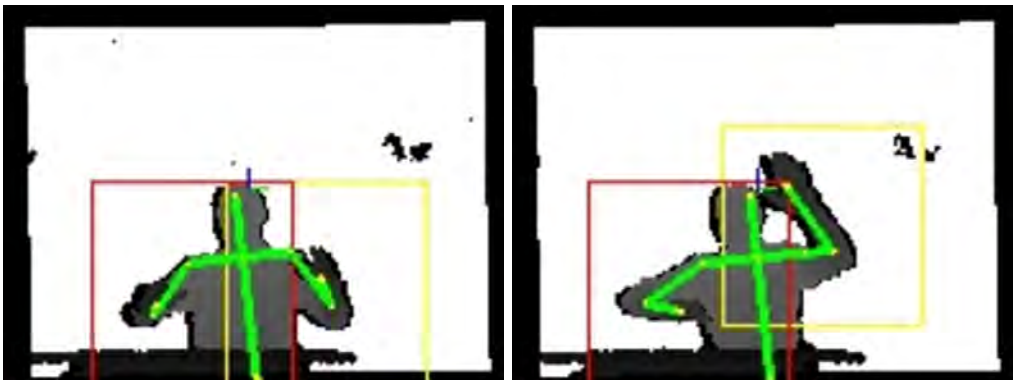


FIGURE B.1: Examples of 2D Bounding Boxes for both right and left arms (overlaid on the input depth data). Only foreground depth pixels enclosed in the respective bounding boxes are used to evaluate the likelihood of each arm particles.

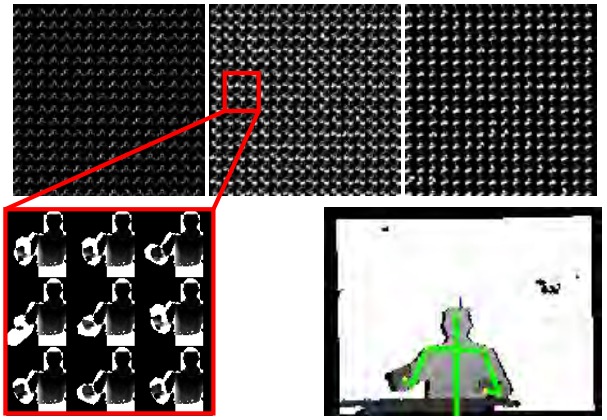


FIGURE B.2: GPU computation of likelihoods. Top Row: Result of the pixel-wise sum of squared differences between depth maps on the *mosaic textures* for the 3 layers involved in the hierarchical evaluation of the likelihood (torso, right arm and left arm). These distance maps are stored in global memory to compute the final particle cost. Bottom Row: Right arm depth cost tiles in detail and input depth data with overlaid final estimation.

where the vertices V are the depth pixels and edges E are defined using 8-connectivity neighborhoods constrained by a maximum distance d_{thr}: if a neighbor pixel represents a point that is farther away than d_{thr} in 3D world coordinates, the edge between both vertices is discarded. We then need to define a source point in order to obtain an approximate geodesic map on the human body surface. We choose the closest pixel to the center of mass of the masked depth information as source point. In fact, we find more convenient to lower the center of mass in order to obtain better geodesic extrema detection. Finally, we filter the geodesic map using a Component Tree and the strategies proposed in [ALMPC11] to obtain the 3 most prominent geodesic extrema, that are likely to accurately represent head and hands location.

These geodesic extrema constitute a set of candidates for hands and head locations at

each time instant t. Provided that the head location is, to a great extent, a result of rigid motion of the torso, we only consider geodesic extrema in the propagation of arm particles.

B.3 Online Initialization

Automated initialization of a body tracker and, specifically, a particle filter-based markerless motion capture system is a challenging problem but also a desired feature for such a technology, as the online usability of the algorithm gets boosted. We propose using the geodesic extrema detection (see Section B.2) and the swing twist IK method [Kal08] under some assumptions to initialize the pose as well as to provide an estimate of the anthropometric profile of the user in terms of bone elongations.

Pose Initialization The assumptions for pose initialization are that the human may stand approximately with his or her back in vertical position, staring in front of the camera, and that right and left hands will be visible to the range sensor. Under such assumptions, whenever three reliable geodesic extrema are detected, we classify them as head, right arm or left arm by tracking the geodesic path that leads to the corresponding geodesic extrema. If the geodesic map has been properly computed, the direction of the paths leading to end-effectors remains similar in the neighborhood of the source point: left hand to the north-east and right hand to the north-west (considering a non-mirrored image) and head is reached through the middle path, usually to the north (see Fig. B.3). Then, we can label the 3 end-effectors depending on the extent to which the geodesic path directions match the *fork* pattern (see Fig. B.3). To match such a pattern, we compute the slope of each path in the neighborhood of the source point. We then check that the relative slope between each pair of paths is within a given interval. To measure the relative slope, we check that the cosinus of the difference angles between each pair of paths is within the interval $[0.5, 0.9]$.

If the geodesic path analysis yields a location for head, right and left hands (namely $\mathbf{g}_0$) then a pose configuration is computed as follows:

1. Translate the model to match the head
2. Compute arm poses by means of IK

a b

FIGURE B.3: Extrema identification during pose initialization: a) Example of the *fork* pattern produced by geodesic paths in the neighborhood of the source point. b) The fork pattern is matched by checking that the cosinus of the angles between paths are in a given interval; then we can identify the three key end-effectors

3. Measure the probability of each computed pose with the following expression

$$\begin{aligned} &\log p(\mathbf{x}_0|\mathbf{g}_0,\mathbf{z}_0) \triangleq \\ &-\frac{1}{2}(\mathbf{g}_0 - F(\mathbf{x}_0))\boldsymbol{\Sigma}_\epsilon^{-1}(\mathbf{g}_t - F(\mathbf{x}_0)) \\ &+\sum_{n=1}^{N}\log \mathcal{U}_{[a_n,b_n]}(\mathbf{x}[n]) + C \\ &-(\lambda'_1 c_d(\mathbf{D}_0,\mathbf{x}_0) + \lambda'_2 c_f(\mathbf{D}_0,\mathbf{x}_0) + \lambda'_3 c_p(\mathbf{x}_0)) \end{aligned} \tag{B.1}$$

4. If $p(\mathbf{x}_0|\mathbf{g}_0,\mathbf{z}_0) > th$, where th is some threshold, accept the initial pose.

where the first term measures the probability of the three end-effector locations and the last term measures the likelihood of the initial pose by means of a sum of squared differences cost function (see Chapter 3 Section 3.6.2). We dropped the layer subindex l, meaning that this expression is calculated for the three partitions. In practice, we average the costs of all the three partitions to obtain the initial probability.

Anthropometric Initialization For anthropometric estimation we assume the same for back inclination and also that, at some point, the target's upper arms will be approximately pointing towards the floor. When this situation happens, we trigger two measurements.

The first one is devoted to estimating the shoulder breadth. This measurement consists in analyzing the derivative of the summation of foreground pixels in each image row. Under the assumptions above, shoulders are found in rows closer to the first prominent local maximum of this derivative (see Fig. B.4). Using this information, we extract

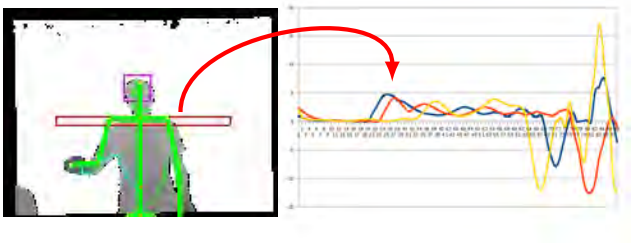
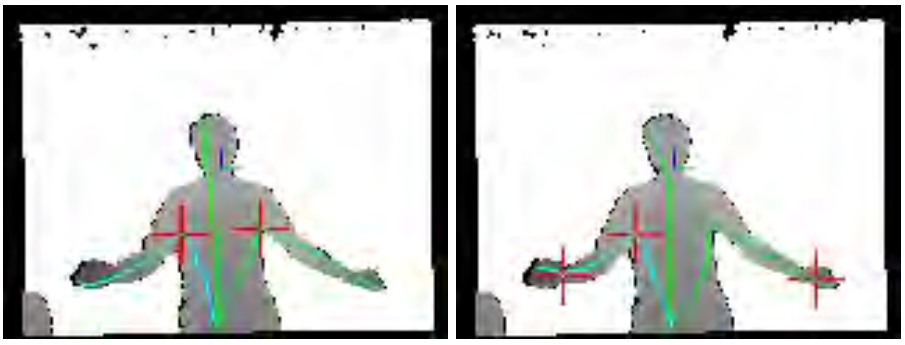


FIGURE B.4: Shoulder breadth estimation: Derivatives of the silhouette width along the vertical axis of the depth image are used to locate the approximate shoulder positions during initialization. Plotted curves, starting at the head top location, show the first prominent local maxima of the derivative for 3 different users. Shoulder pixels are located in the neighborhood of this first prominent local maxima of the derivative.



a b

FIGURE B.5: Points with high curvature along the projected geodesic path are marked with a red-cross. a) Correct detections below both shoulders b) Spurious detections near wrists. These detections are automatically rejected due to the proximity to the end-effectors.

several points from the leftmost and rightmost pixels of the rows around this maximum to compute an approximation of the shoulder breadth.

The second measurement concerns the arm length and is computed through the analysis of the geodesic path lengths. In order to remove parts of the path located in the torso, we use second derivatives to analyze the curvature of the projection of the geodesic path on the image (see Fig. B.5). Points with high curvature located between 25% and 50% of the path length (in real world units and starting from the source point) usually fall right below the shoulder joint, allowing us to use the resting path length as arm length measurement. If the arm length obtained by this method is within an interval of possible human arm lengths, the measurement is collected as valid. The total arm length is split into upper and forearm based on anthropometric studies [nas87].

In both cases, we perform anthropometric measurements during a short period of time. We perform outlier rejection by analyzing the variance of the collected measurements to finally provide the bone elongations needed to approximately fit the upper body model to the user.

Tile Size	Device	64x3	256x3	1024x3
64x64	NVIDIA Quadro FX 3700	11.2	3.3	0.9
64x64	NVIDIA GeForce GTX 295	13.2	4.5	1.3
128x128	NVIDIA Quadro FX 3700	3.1	1.6	0.5
128x128	NVIDIA GeForce GTX 295	3.9	2.2	0.8
256x256	NVIDIA Quadro FX 3700	0.8	0.1	-
256x256	NVIDIA GeForce GTX 295	1.0	0.2	0.1

TABLE B.1: Computational performance of the complete system (in frames per second) as a function of the number of particles (x3 layers) and the size of the tiles for different hardware platforms. The size of the tiles is proportional to the image resolution (e.g. a 64x64 tile is for 160x120 depth maps). *Mosaic textures* of 8192x8192 are not supported by the NVIDIA Quadro FX 3700.

The proposed method for estimating the anthropometric profile of the user barely relies on the model and the estimated pose. As a consequence, the robustness of the anthropometric estimates increases, because it is not affected by pose estimation errors.

B.4 Experimental Results

To validate our algorithm, we have conducted several experiments with range data recorded with a Kinect (640x480 pixels, 30fps). We consider two different scenarios where upper body motion is involved: desktop (Fig. B.6) and workplace (Fig. B.7). In desktop, users are sitting down in front of a table, while in workplace are standing up.

Desktop sequences involve 5 users performing several actions such as motion of one arm, picking a phone, or drawing some figure with both hands. These sequences comprise almost 4 minutes of data. Workplace sequences contain approximately 1 minute and a half of challenging motions performed by 2 subjects.

We pick several subsequences from this data and we manually annotate pixels belonging to joint positions. We perform these annotations in 1 of every 10 frames, obtaining around 470 annotated frames and more than 4700 frames to evaluate.

In the first experiment, we evaluate the computational performance of the implemented system in two different hardware platforms: both CPUs are Intel 2.80GHz, 4GB RAM, but different GPUs are installed (see Table B.1). Although we have only optimized the likelihood computation, the system runs online in a laptop with an ATI Mobility Radeon HD5800; to reach the online performance, we use a total of 192 particles and 160x120 depth maps (the original data at 1/16 resolution). With this configuration, we obtain a satisfactory performance for several upper body actions.

In the second experiment, we run our system offline with the recorded sequences in order to compare our results with the recently released PrimeSense body tracker for Kinect,

FIGURE B.6: Tracking results in desktop upper-body sequences recorded with Kinect. 256 particles per layer are used.

FIGURE B.7: Tracking results in workplace upper-body sequences recorded with Kinect. 256 particles per layer are used.

which is accessible thanks to the OpenNI middleware [Pri]. This tracker requires a specific initial pose (hands up) called the *calibration pose*. In order to perform a comparison between both methods, all the annotated sequences have been recorded with this *calibration pose*. Nonetheless, since our method incorporates automated initialization of pose and anthropometrics, we can successfully launch our tracker without requiring such a specific pose. In these experiments, we use 256 particles in each hierarchical layer and depth maps of 160x120 pixels, yielding close to real-time performance.

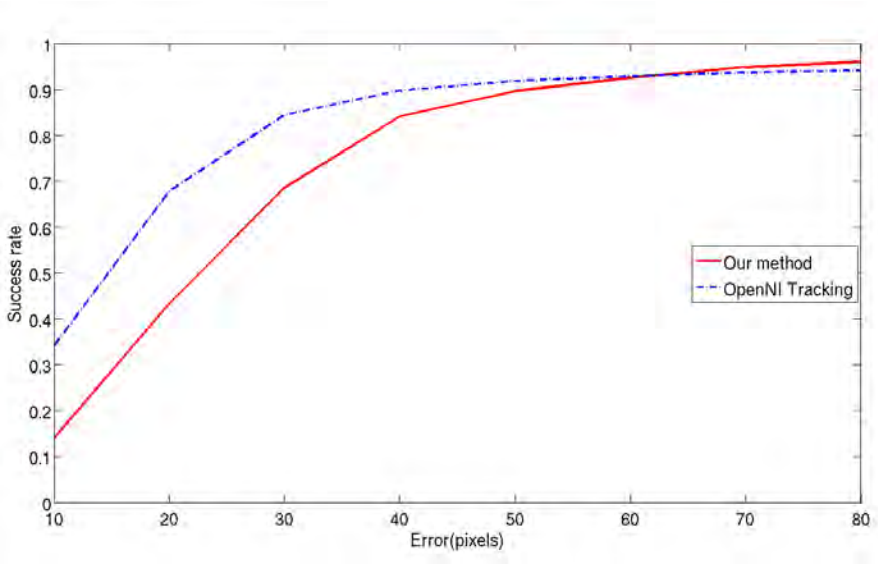


FIGURE B.8: Success rate comparative between PrimeSense tracker and our method. Errors have been computed in 640x480 images.

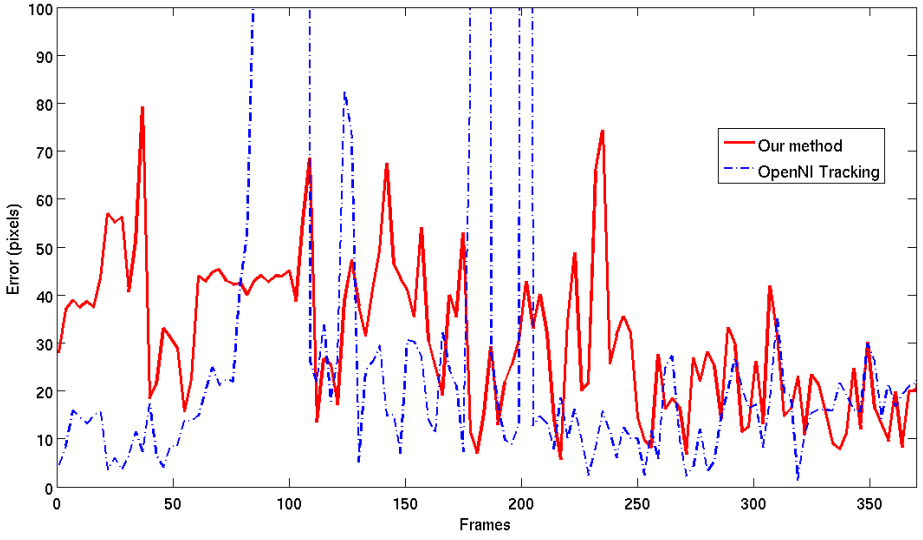


FIGURE B.9: Right hand error (pixels) for the PrimeSense tracker and our method in a sequence where PrimeSense tracker has several misses. Errors have been computed in 640x480 images.

Using the available annotations, we measure the accuracy/precision of both systems by means of a success rate: given a distance in pixels, we count the percentage of joints that have been tracked with an error below this distance. In the annotated sequences, hands are visible all the time, hence both algorithms should be able to provide estimates for the upper body joints. Since PrimeSense tracker may not provide joint locations that considers unreliable, we consider these cases as misses. The results, measured in 640x480 frames, are shown in Fig. B.8. Our system shows a remarkable precision, although being less accurate than the PrimeSense tracker, that relies on accurate part

detectors. However, we have observed that, while for a long tracking period and fast motions, the PrimeSense tracker outperforms our method, there are some cases, with limb self-occlusions, in which our method has a better performance. Specifically, when shoulders are occluded or a hand is partially occluded, the PrimeSense tracker can fail (see Fig. B.9). These cases are more frequent in the desktop scenario, showing that the upper body tracking task is a difficult problem due to important occlusions of the lower body. In our method, the use of a body model and hierarchical layers helps in overcoming these cases. In overall, the PrimeSense method presents a mean tracking error of 20.74 pixels (excluding misses) while ours has an error of 28.95 pixels. The mean initialization error of our method is 33.31 pixels.

Bibliography

[ABS07] S. Ali, A. Basharat, and M. Shah. Chaotic invariants for human action recognition. In *ICCV*, pages 1–8, Oct. 2007.

[ACP10] M. Alcoverro, J. Casas, and M. Pardàs. Skeleton and Shape Adjustment and Tracking in Multicamera Environments. In Francisco J Perales López and Robert B Fisher, editors, *AMDO*, volume 6169 of *Lecture Notes in Computer Science*, pages 88–97. Springer, 2010.

[ALMCP11] M. Alcoverro, A. Lopez-Mendez, J. R. Casas, and M. Pardas. A real-time body tracking system for smart rooms. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1 –6, july 2011.

[ALMPC11] M. Alcoverro, A. Lopez-Mendez, M. Pardas, and J.R. Casas. Connected operators on 3d data for human body analysis. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 9 –14, june 2011.

[AMG+02] MS Arulampalam, S. Maskell, N. Gordon, T. Clapp, D. Sci, T. Organ, and SA Adelaide. A tutorial on particle filters for online nonlinear/non-GaussianBayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[AT06] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(1):44 –58, jan. 2006.

[BB09] J. Bandouch and M. Beetz. Tracking humans interacting with the environment using efficient hierarchical sampling and layered observation models. In *ICCV-HCI*, 2009.

[BC96] Donald J. Berndt and James Clifford. Advances in knowledge discovery and data mining. chapter Finding patterns in time series: a dynamic programming approach, pages 229–248. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[BC08] Luca Ballan and Guido Maria Cortelazzo. Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. In *3DPVT*, USA, 2008.

[BCMCB09] Olivier Bernier, Pascal Cheung-Mon-Chan, and Arnaud Bouguet. Fast nonparametric belief propagation for real-time stereo articulated body tracking. *CVIU*, 113(1):29–47, 2009.

[BCMS01] A. Bissacco, A. Chiuso, Yi Ma, and S. Soatto. Recognition of human gaits. In *CVPR*, volume 2, pages II–52 – II–57 vol.2, 2001.

[BD01] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, Mar 2001.

[BEB08] Jan Bandouch, Florian Engstler, and Michael Beetz. Evaluation of hierarchical sampling strategies in 3d human pose estimation. In *BMVC*, 2008.

[BJB12] Jan Bandouch, Odest Jenkins, and Michael Beetz. A self-training approach for visual tracking and recognition of complex human activity patterns. *International Journal of Computer Vision*, 99:166–189, 2012. 10.1007/s11263-012-0522-y.

[BKSS10] Martin Bergtholdt, Jrg Kappes, Stefan Schmidt, and Christoph Schnrr. A study of parts-based object class detection using complete graphs. *International Journal of Computer Vision*, 87:93–117, 2010. 10.1007/s11263-009-0209-1.

[BM98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *CVPR*, pages 8 –15, jun 1998.

[BP07] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. *ACM Trans. Graph.*, 26(3), July 2007.

[BR07] Jaron Blackburn and Eraldo Ribeiro. Human motion recognition using Isomap and Dynamic Time Warping. In *Proceedings of the 2nd conference on Human motion: understanding, modeling, capture and animation*, pages 285–298, Berlin, Heidelberg, 2007. Springer-Verlag.

[Bre01] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[BS09] A. Basharat and M. Shah. Time series prediction by chaotic modeling of nonlinear dynamical systems. In *ICCV 2009*, pages 1941–1948, 2009.

[BSP+04] J. Barbič, A. Safonova, J. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface 2004*, GI '04, pages 185–194, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

[Cao97] Liangyue Cao. Practical method for determining the minimum embedding dimension of a scalar time series. *Physica D: Nonlinear Phenomena*, 110(1-2):43 – 50, 1997.

[Car] Carnegie Mellon University Motion Capture Database. http://mocap.cs.cmu.edu.

[CB95] L. Campbell and A. Bobick. Recognition of human body motion using phase space constraints. In *ICCV*, pages 624–630, 1995.

[CCPM09] R. Cabido, D. Concha, J.J. Pantrigo, and A.S. Montemayor. High Speed Articulated Object Tracking Using GPUs: A Particle Filter Approach. In *ISPAN*, pages 757 –762, dec. 2009.

[CF10] C. Canton-Ferrer. *Human Motion Capture with Scalable Body Models*. PhD thesis, Technical University of Catalonia (UPC), 2010.

[CFCP06] C. Canton-Ferrer, J. R. Casas, and M. Pardàs. Human model and motion based 3d action recognition in multiple view scenarios (invited paper). In *14th European Signal Processing Conference, EUSIPCO*, 2006. ISBN: 0-387-34223-0.

[CFCP11] Cristian Canton-Ferrer, Josep R. Casas, and Montse Pards. Human motion capture using scalable body models. *Computer Vision and Image Understanding*, 115(10):1363 – 1374, 2011.

[CMG+10] Stefano Corazza, Lars Mndermann, Emiliano Gambaretto, Giancarlo Ferrigno, and Thomas Andriacchi. Markerless motion capture through visual hull, articulated icp and subject specific model generation. *International Journal of Computer Vision*, 87:156–169, 2010. 10.1007/s11263-009-0284-3.

[CMJ98] Liangyue Cao, Alistair Mees, and Kevin Judd. Dynamics from multivariate time series. *Phys. D*, 121(1-2):75–88, October 1998.

[CSK12] Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Found. Trends. Comput. Graph. Vis.*, 7(2–3):81–227, February 2012.

[CWSS07] Tat-Jun Chin, Liang Wang, K. Schindler, and D. Suter. Extrapolating learned manifolds for human activity recognition. In *ICIP*, volume 1, pages I –381 –I –384, 16 2007-oct. 19 2007.

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

[CZXL09] C. Chen, Y. Zhuang, J. Xiao, and Z. Liang. Perceptual 3d pose distance estimation by boosting relational geometric features. *Comput. Animat. Virtual Worlds*, 20(23):267–277, jun 2009.

[DBR00] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, 2:126–133 vol.2, 2000.

[DD09] Yuanqiang Dong and G.N. DeSouza. A new hierarchical particle filtering for markerless human motion capture. In *Computational Intelligence for Visual Intelligence, 2009. CIVI '09. IEEE Workshop on*, pages 14 –21, 30 2009-april 2 2009.

[DGA00] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000.

[DKJ+07] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 209–216, New York, NY, USA, 2007. ACM.

[DR05] Jonathan Deutscher and Ian Reid. Articulated body motion capture by stochastic search. *Int. J. Comput. Vision*, 61(2):185–205, 2005.

[DRCB05] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 65 – 72, oct. 2005.

[DSTT00] F. Dellaert, S.M. Seitz, C.E. Thorpe, and S. Thrun. Structure from motion without correspondence. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 557 –564 vol.2, 2000.

[EL04] A. Elgammal and Chan-Su Lee. Inferring 3d body pose from silhouettes using activity manifold learning. *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2:II–681–II–688 Vol.2, June-2 July 2004.

[Eur] European Project FascinatE. http://www.fascinate-project.eu.

[FBF77] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3:209–226, September 1977.

[FP7] FP7 Project ACTIBIO. http://www.actibio.eu.

[FS86] A. M. Fraser and H. L. Swinney. Independent coordinates for strange attractors from mutual information. *Physical Review A*, 33(2):1134–1140, Feb 1986.

[FSJW08] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. Nonparametric Bayesian learning of switching linear dynamical systems. In *NIPS*. MIT Press, 2008.

[Fut] Future Light R&D division of Santa Monica.

[GBS+07] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, December 2007.

[GFA12] Gutemberg Guerra-Filho and Yiannis Aloimonos. The syntax of human actions and interactions. *Journal of Neurolinguistics*, 25(5):500 – 514, 2012.

[GPS+07] Jürgen Gall, Jürgen Potthoff, Christoph Schnörr, Bodo Rosenhahn, and Hans-Peter Seidel. Interacting and annealing particle filters: Mathematics and a recipe for applications. *J. Math. Imaging Vis.*, 28:1–18, May 2007.

[GRBS10] J. Gall, B. Rosenhahn, T. Brox, and H. Seidel. Optimization and filtering for human motion capture. *International Journal of Computer Vision*, 87:75–92, 2010. 10.1007/s11263-008-0173-1.

[GRS08] J. Gall, B. Rosenhahn, and H.-P. Seidel. Drift-free tracking of rigid and articulated objects. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, june 2008.

[GSS93] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113, Apr 1993.

[GYR+11] J. Gall, A. Yao, N. Razavi, L. van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *TPAMI*, 33(11):2188 –2202, nov. 2011.

[GYvG10] Juergen Gall, Angela Yao, and Luc van Gool. 2d action recognition serves 3d human pose estimation. In *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*, ECCV'10, pages 425–438, Berlin, Heidelberg, 2010. Springer-Verlag.

[Hal03] B. Hall. Lie Groups, Lie Algebras, and Representations: An Elementary Introduction (Graduate Texts in Mathematics vol 222), 2003.

[HDM] HDM05 Mocap Dataset. http://www.mpi-inf.mpg.de/resources/hdm05/index.html.

[HKKR93] Daniel P. Huttenlocher, Gregory A. Klanderman, Gregory A. Kl, and William J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.

[How04] Nicholas R. Howe. Silhouette lookup for automatic pose tracking. *Computer Vision and Pattern Recognition Workshop*, 1:15–22, 2004.

[HP11] Søren Hauberg and Kim Pedersen. Predicting articulated human motion from spatial processes. *IJCV*, pages 1–18, 2011.

[HTTM11] Michael B. Holte, Cuong Tran, Mohan M. Trivedi, and Thomas B. Moeslund. Human action recognition using multiple views: a comparative perspective on recent developments. In *ACM workshop on HGBU*, J-HGBU '11, pages 47–52, New York, NY, USA, 2011. ACM.

[HWG07] Zsolt Husz, Andrew M. Wallace, and Patrick R. Green. Evaluation of a hierarchical partitioned particle filter with action primitives. In *2-nd Workshop on Evaluation of Articulated Human Motion and Pose Estimation (EHUM2), CVPR*. IEEE, 2007.

[HWG11] Z.L. Husz, A.M. Wallace, and P.R. Green. Tracking with a hierarchical partitioned particle filter and movement modelling. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(6):1571 – 1584, dec. 2011.

[IB98] M. Isard and A. Blake. CONDENSATION-Conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1):5–28, 1998.

[Inr06] Inria. The IXMAS Dataset. http://4drepository.inrialpes.fr/public/viewgroup/6, 2006.

[JKMvG09] Tobias Jaeggli, Esther Koller-Meier, and Luc van Gool. Learning generative models for multi-activity body pose estimation. *Int. J. Comput. Vision*, 83(2):121–134, June 2009.

[JLD12] Zhuolin Jiang, Zhe Lin, and L.S. Davis. Recognizing human actions by learning and matching shape-motion prototype trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(3):533 –547, march 2012.

[Joh73] G Johansson. Visual perception of biological motion and a model for its analysis. *Perception And Psychophysics*, 14(2):201–211, 1973.

[JR99] M.J. Jones and J.M. Rehg. Statistical color models with application to skin detection. In *CVPR*, volume 1, pages 2 vol. (xxiii+637+663), 1999.

[JSWP07] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, pages 1 –8, oct. 2007.

[Kal08] Marcelo Kallmann. Analytical inverse kinematics with body posture control. *Journal of Visualization and Computer Animation*, 19:79–91, 2008.

[KBA92] M. B. Kennel, R. Brown, and H. D. I. Abarbanel. Determining embedding dimension for phase-space reconstruction using a geometrical construction. *Phys. Rev. A*, 45(6):3403–3411, Mar 1992.

[KBvGF10] D. Kuettel, M.D. Breitenstein, L. van Gool, and V. Ferrari. What's going on? discovering spatio-temporal dependencies in dynamic scenes. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1951 –1958, june 2010.

[KG10] Adriana Kovashka and Kristen Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR*, 2010.

[KGHW11] O. Kliper-Gross, T. Hassner, and L. Wolf. One shot similarity metric learning for action recognition. In *Proceedings of the First international*

conference on Similarity-based pattern recognition, SIMBAD'11, pages 31–45, Berlin, Heidelberg, 2011. Springer-Verlag.

[KH09] Hansung Kim and Adrian Hilton. Graph-based foreground extraction in extended color space. In *Proceedings of the 16th IEEE international conference on Image processing*, ICIP'09, pages 3185–3188, Piscataway, NJ, USA, 2009. IEEE Press.

[Kin] Microsoft Kinect. http://www.xbox.com/kinect.

[KL03] Eamonn Keogh and Jessica Lin. Clustering of time series subsequences is meaningless: Implications for past and future research. In *In Proc. of the 3rd IEEE International Conference on Data Mining*, pages 115–122, 2003.

[KMS08] Alexander Kläser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *British Machine Vision Conference*, pages 995–1004, sep 2008.

[KOSS11] K.M. Kitani, T. Okabe, Y. Sato, and A. Sugimoto. Fast unsupervised ego-action learning for first-person sports videos. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3241 –3248, june 2011.

[KS04] H. Kantz and T. Schreiber. Nonlinear time series analysis. Cambridge U. Press, 2004.

[KT08] Reinhard Klette and Garry Tee. Understanding human motion : A historic review. *Classical Antiquity*, pages 1–21, 2008.

[LAaKR10] N.H. Lehment, D. Arsic and, M. Kaiser, and G. Rigoll. Automated pose estimation in 3D point clouds applying annealing particle filters and inverse kinematics on a GPU. In *CVPR Workshops*, pages 87 –92, june 2010.

[LAM+11] Binlong Li, M. Ayazoglu, T. Mao, O.I. Camps, and M. Sznaier. Activity recognition using dynamic subspace angles. In *CVPR*, pages 3193 –3200, june 2011.

[Law04] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *In NIPS*, page 2004, 2004.

[LC98] J.S. Liu and R. Chen. Sequential Monte Carlo methods for dynamical systems. *Journal of the American Statistical Association*, 93(5):1032–1044, 1998.

[LE10] C. S. Lee and A. Elgammal. Coupled visual and kinematic manifold models for tracking. *Int. J. Comput. Vision*, 87(1-2):118–139, March 2010.

[LL03] Ivan Laptev and Tony Lindeberg. Space-time interest points. In *In ICCV*, pages 432–439, 2003.

[LMAPC11a] A. Lopez-Mendez, M. Alcoverro, M. Pardas, and J.R. Casas. Approximate partitioning of observations in hierarchical particle filter body tracking. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 19 –24, june 2011.

[LMAPC11b] A. Lopez-Mendez, M. Alcoverro, M. Pardas, and J.R. Casas. Real-time upper body tracking with online initialization using a range sensor. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 391 –398, nov. 2011.

[LMC12a] A. López-Méndez and J.R. Casas. Can our tv robustly robustly understand human gestures? *European Conference on Visual Media Production*, To appear 2012.

[LMC12b] A. López-Méndez and J.R. Casas. Model-based recognition of human actions by trajectory matching in phase spaces. *Image and Vision Computing*, (0):–, 2012.

[LMCFC10] A. López-Méndez, C. Canton-Ferrer, and J.R. Casas. Virtual view appearance representation for human motion analysis in multi-view environments. *In Proc. of European Signal Processing Conference (EUSIPCO),*, pages 959–963, August 2010.

[LMCvGL12] J. López-Méndez, A. Gall, J.R. Casas, and van Gool L. Metric learning from poses for temporal clustering of human motion. In *Proceedings of the British Machine Vision Conference*, pages 49.1–49.12. BMVA Press, 2012.

[LMdRMN10] M. Lewandowski, J. Martinez-del Rincon, D. Makris, and J.-C. Nebel. Temporal extension of laplacian eigenmaps for unsupervised dimensionality reduction of time series. In *ICPR*, pages 161 –164, aug. 2010.

[LMS+] Ivan Laptev, Marcin Marszaek, Cordelia Schmid, Benjamin Rozenfeld, Inria Rennes, Irisa Inria Grenoble, and Lear Ljk. B.: Learning realistic human actions from movies. In *In: CVPR. (2008.*

[LNL05] F. Lv, R. Nevatia, and M. W. Lee. 3d human action recognition using spatio-temporal motion templates. In *ICCV-HCI*, pages 120–130, 2005.

[LTSY10] Rui Li, Tai-Peng Tian, Stan Sclaroff, and Ming-Hsuan Yang. 3d human motion tracking with a coordinated mixture of factor analyzers. *Int. J. Comput. Vision*, 87(1-2):170–190, March 2010.

[Mac03] D.J.C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[Mar94] E.-J. Marey. *Le Mouvement*. G. Masson, Paris, 1894.

[MH03] J. Mitchelson and A. Hilton. Simultaneous pose estimation of multiple people using multiple-view cues with hierarchical sampling. *In Proc. of BMVC, September*, 2003.

[MHK06] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *CVIU*, 104(2-3):90–126, 2006.

[MI00] J. MacCormick and M. Isard. Partitioned Sampling, Articulated Objects, and Interface-Quality Hand Tracking. *Lecture Notes in Computer Science*, 1843:3–19, 2000.

[MRC05] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. *ACM Transactions on Graphics (SIGGRAPH)*, 24(3):677–685, July 2005.

[nas87] National Aeronautics And Space Administration. Man-Systems Integration Standards. Technical report, 1987.

[NLMAC12] Sergio Navarro, A. López-Méndez, M. Alcoverro, and J. Casas. *Multi-view Body Tracking with a Detector-Driven Hierarchical Particle Filter*, volume 7378 of *Lecture Notes in Computer Science*, pages 82–91. Springer, Berlin / Heidelberg, 2012.

[NWFf06] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-fei. Unsupervised learning of human action categories using spatial-temporal words. In *In Proc. BMVC*, 2006.

[OMB09] Björn Ommer, Theodor Mader, and Joachim M. Buhmann. Seeing the objects behind the dots: Recognition in videos from a moving camera. *Int. J. Comput. Vision*, 83(1):57–71, June 2009.

[OSC08] N. Ozay, M. Sznaier, and O.I. Camps. Sequential sparsification for change detection. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –6, june 2008.

[Par94] F.C. Park. Computational aspects of the product-of-exponentials formula for robot kinematics. *Automatic Control, IEEE Transactions on*, 39(3):643–647, 1994.

[PC06] V. Parameswaran and R. Chellappa. View invariance for human action recognition. *Int. Journal of Computer Vision*, 66:83–101, January 2006.

[Pic04] M. Piccardi. Background subtraction techniques: a review. In *2004 IEEE Conference on Systems, Man and Cybernetics*, volume 4, 2004.

[Pop10] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[Pow64] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, January 1964.

[PR00] V. Pavlovic and J.M. Rehg. Impact of dynamic model learning on classification of human motion. In *CVPR*, volume 1, pages 788 –795 vol.1, 2000.

[Pri] PrimeSense. http://www.primesense.com/.

[PVW10] Patrick Peursum, Svetha Venkatesh, and Geoff West. A study on smoothing for particle-filtered 3d human body tracking. *Int. J. Comput. Vision*, 87(1-2):53–74, March 2010.

[RBS07] Michalis Raptis, Matteo Bustreo, and Stefano Soatto. Time warping under dynamic constraints. 2007.

[RRR09] Leonid M. Raskin, Michael Rudzsky, and Ehud Rivlin. 3d human body-part tracking and action classification using a hierarchical body model. In *BMVC*, 2009.

[RS00] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[Rus09] Radu Bogdan Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, October 2009.

[RWS08] M. Raptis, K. Wnuk, and S. Soatto. Flexible dictionaries for action classification. In *Proceedings of the International Workshop on Machine Learning for Vision-based Motion Analysis, in conjunction with ECCV*, 2008.

[RWS10] M. Raptis, K. Wnuk, and S. Soatto. Spike train driven dynamical models for human actions. In *CVPR*, pages 2077–2084, 2010.

[SALM+ss] X. Suau, M. Alcoverro, A. López-Méndez, J. Ruiz-Hidalgo, and J. Casas. *INTAIRACT: Joint Hand Gesture and Fingertip Classification for Touchless Interaction*, volume 7585, chapter 3, pages 602–606. Springer, Heidelberg, In Press.

[SAS07] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM.

[SBB10] L. Sigal, A. O. Balan, and M. J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *IJCV*, 87(1-2):4–27, 2010.

[SBF00] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, ECCV '00, pages 702–718, London, UK, UK, 2000. Springer-Verlag.

[SBR+04] L. Sigal, S. Bhatia, S. Roth, M.J. Black, and M. Isard. Tracking loose-limbed people. *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 1:I–421–I–428 Vol.1, June-2 July 2004.

[SBS02] Hedvig Sidenbladh, Michael J. Black, and Leonid Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, pages 784–800, London, UK, UK, 2002. Springer-Verlag.

[SC78] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43 – 49, feb 1978.

[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 *Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304, Colorado Springs, 2011. IEEE.

[SG00] C. Stauffer and W.E.L. Grimson. Learning patterns of activity using real-time tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):747 –757, aug 2000.

[SJC08] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR 2008*, pages 1 –8, june 2008.

[SM10] M. Siddiqui and G. Medioni. Human pose estimation from a single view point, real-time range sensor. In *CVPR Workshops*, pages 1 –8, june 2010.

[SNH10] Vivek Kumar Singh, Ram Nevatia, and Chang Huang. Efficient inference with multiple heterogeneous part detectors for human pose estimation. In *ECCV*, ECCV'10, pages 314–327, Berlin, Heidelberg, 2010. Springer-Verlag.

[SSC10] Jordi Salvador, Xavier Suau, and Josep R. Casas. From silhouettes to 3d points to mesh: towards free viewpoint video. In *Proceedings of the 1st international workshop on 3D video processing*, 3DVP '10, pages 19–24, 2010.

[SvG08] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, june 2008.

[SWFS03] M. Seki, T. Wada, H. Fujiwara, and K. Sumi. Background subtraction based on cooccurrence of image variations. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–65 – II–72 vol.2, june 2003.

[Tak81] F. Takens. Detecting strange attractors in turbulence. *Dynamical Systems and Turbulence*, pages 366–381, 1981.

[TCMS03] Christian Theobalt, Joel Carranza, Marcus A. Magnor, and Hans-Peter Seidel. A parallel framework for silhouette-based human motion capture. In *In Proc. VMV*, pages 207–214. DNB, 2003.

[TCSU08] P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, Nov. 2008.

[TJBB06] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[TLKS08] J. K. T. Tang, H. Leung, T. Komura, and H. P. H. Shum. Emulating human perception of motion similarity. *Comput. Animat. Virtual Worlds*, 19(3-4):211–221, sep 2008.

[TNI+09] R. Tsuchiyama, T. Nakamura, T. Iizuka, A. Asahara, and S. Miki. The OpenCL Programming Book. *Group*, 2009.

[TS08] D. Tran and A. Sorokin. Human activity recognition with metric learning. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 548–561, Berlin, Heidelberg, 2008. Springer-Verlag.

[TSFH10] G.W. Taylor, L. Sigal, D.J. Fleet, and G.E. Hinton. Dynamical binary latent variable models for 3d human pose tracking. In *CVPR*, pages 631 –638, june 2010.

[TSL00] J.B. Tenenbaum, V. Silva, and J.C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, 2000.

[UD08] R. Urtasun and T. Darrell. Sparse probabilistic regression for activity-independent human pose inference. In *CVPR*, pages 1 –8, june 2008.

[UFF06] R. Urtasun, D.J. Fleet, and P. Fua. 3d people tracking with gaussian process dynamical models. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 1:238–245, June 2006.

[UFHF05] R. Urtasun, D.J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 403 – 410 Vol. 1, oct. 2005.

[Vap95] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[WADP97] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfinder: real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780 –785, jul 1997.

[War63] Jr. Ward. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 58:236–244, 1963.

[WB03] J. Wang and B. Bodenheimer. An evaluation of a cost metric for selecting transitions between motion segments. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, pages 232–238, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[WBR07] Daniel Weinland, Edmond Boyer, and Remi Ronfard. Action recognition from arbitrary views using 3d exemplars. In *Proceedings of the International Conference on Computer Vision, Rio de Janeiro, Brazil*, pages 1–7. IEEE Computer Society Press, 2007.

[WFH05] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models. In *Advances in Neural Information Processing Systems (NIPS) 18*, pages 1441–1448, Vancouver, Canada, December 2005. MIT Press.

[WFH08] J.M. Wang, D.J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283 –298, feb. 2008.

[WKSL11] Heng Wang, A. Klaser, C. Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169 –3176, june 2011.

[WOS11] Shandong Wu, O. Oreifej, and M. Shah. Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1419 –1426, nov. 2011.

[WS07] Liang Wang and D. Suter. Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. In *CVPR*, pages 1 –8, june 2007.

[WS09] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, June 2009.

[WTvG08] Geert Willems, Tinne Tuytelaars, and Luc van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of the 10th European Conference on Computer Vision: Part II*, ECCV '08, pages 650–663, Berlin, Heidelberg, 2008. Springer-Verlag.

[XLP05] L.Q. Xu, JL Landabaso, and M. Pardas. Shadow removal with blob-based morphological reconstruction for error correction. In *ICASSP'05*, volume 2. IEEE, 2005.

[Xti] ASUS Xtion. http://www.asus.com/multimedia/motion_sensor/xtion_pro.

[YGFvG11] Angela Yao, Juergen Gall, Gabriele Fanelli, and Luc van Gool. Does human action recognition benefit from pose estimation? In *Proc. BMVC*, pages 67.1–67.11, 2011. http://dx.doi.org/10.5244/C.25.67.

[YGvG12] Angela Yao, Juergen Gall, and Luc van Gool. Coupled action recognition and pose estimation from multiple views. *International Journal of Computer Vision*, 100:16–37, 2012. 10.1007/s11263-012-0532-9.

[YW09] L. Yeffet and L. Wolf. Local trinary patterns for human action recognition. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 492 –497, 29 2009-oct. 2 2009.

[ZDH08] F. Zhou, F. De la Torre, and J. K. Hodgins. Aligned Cluster Analysis for Temporal Segmentation of Human Motion. In *IEEE Conference on Automatic Face and Gestures Recognition (FG)*, September 2008.

[ZDH11] F. Zhou, F. De la Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *Under review at IEEE PAMI*, 2011.