UNIVERSITY OF CALIFORNIA
Los Angeles

Understanding the Spatio-Temporal Structure of Human Actions

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Computer Science

by

Michail Raptis

2011
The dissertation of Michail Raptis is approved.

________________________________________________________________________
Alan L. Yuille

________________________________________________________________________
Demetri Terzopoulos

________________________________________________________________________
Paulo Tabuada

________________________________________________________________________
Stefano Soatto, Committee Chair

University of California, Los Angeles
2011
To Popi, Giorgos and Nikoletta, my family
# TABLE OF CONTENTS

1 **Introduction** ............................................................ 1

1.1 Problem Statement and Challenges .................................. 3

1.2 Thesis Outline .............................................................. 6

2 **Background** ............................................................... 9

2.1 Holistic representations ................................................ 9

2.2 Local feature based methods ........................................... 12

2.3 Part based representations .............................................. 16

3 **Flexible Dictionaries** .................................................. 19

3.1 Introduction ................................................................. 19

3.2 Actions as Time Series .................................................. 21

3.3 Dictionary of Action Primitives ......................................... 22

3.3.1 Construction of a Simple Dictionary .............................. 22

3.3.2 Increasing Representative Power by Mean Quantization .... 23

3.3.3 Modeling Time Scale for Temporal Flexibility ............... 24

3.4 Classification ............................................................... 27

3.4.1 Characterizing an Action ............................................. 27

3.4.2 Bagging Action Primitives .......................................... 27

3.4.3 Exploiting Temporal Constraints .................................. 28

3.5 Experiments ............................................................... 30

3.5.1 Datasets ................................................................. 30
A.4 Experiments .............................................................. 138
A.5 Discussion .............................................................. 144

References ................................................................. 151
LIST OF FIGURES

1.1 Examples of human activities captured in video sequences. . . . . . . . . . 3

3.1 The action primitive dictionaries constructed without, Fig. 3.1a, and with, Fig. 3.1b, mean quantization for the $x$-coordinate of the left hand in the FutureLight motion capture dataset. . . . . . . . . . . . . . . . 24

3.2 Reconstruction of a time series representing a coordinate of the relative position of one of the limbs using the best matching dictionary elements. 26

4.1 This figure summarizes how our model captures actions and compares the representational power of sparse input driven LDS with that of traditional stochastic LDS for non-stationary actions. . . . . . . . . . . . . 44

4.2 A spinning ballet dance sequence. The original sequence is shown in the top figure, and the synthetic sequence generated from our sparse input LDS is shown in the bottom figure. . . . . . . . . . . . . . . . . 50

4.3 This figure shows the performance of our temporal segmentation/action recognition on complex CMU MoCap data. . . . . . . . . . . . . . . . . 55

5.1 Flowchart of the gesture classification system. . . . . . . . . . . . . 58

5.2 Skeletal representation. . . . . . . . . . . . . . . . . . . . . . . . . . 61

5.3 Comparison of absolute and relative motion patterns of joints during the dance gesture “ClapSlideNStamp”, as rendered in the 3D coordinate frame of the sensor. . . . . . . . . . . . . . . . . . . . . . . . . . . . 63

5.4 Illustration of torso PCA frame, and its use in defining spherical coordinate systems for first- and second-degree joints. . . . . . . . . . . . . 64
5.5 Subset of feature vectors for an instance of a subject dancing “Clap-SlideNStomp” ................. 66
5.6 Feature vectors for the derived prototype mean and the oracle that correspond to the “ClapSlideNStomp” gesture. .......... 69
5.7 Histograms $p_i$ and $q_i$ obtained over the training set of recordings for the “ClapSlideNStomp” gesture. .......... 72

6.1 Tracks extracted from ADL, KTH and HOHA datasets. Color indicates their label based to the tracklet descriptor dictionary. .......... 93
6.2 A track with samples of the histogram of gradient orientation and histogram of optical flow along the trajectory. ............... 94
6.3 Example of the tracks and an instance of the corresponding appearance descriptor of a boxing action on the KTH dataset. .......... 96
6.4 Confusion matrices for ADL dataset and for KTH dataset. .......... 102

7.1 Examples of trajectory groups; each group has a distinct color. .......... 107
7.2 Localization scores for the trajectory groups selected by our algorithm as function of the overlap threshold ($\theta$). .......... 117
7.3 Sample frames from different video sequences of the test set across different actions. The colored trajectories represent selected trajectory groups identified by our algorithm. .......... 119

A.1 The result of the proposed optical flow approach on two sequences. .......... 140
A.2 This figure illustrates the initial estimate of the error term $e_1$ and how sparsity of $|We_1|$ improves with each of the three re-weighting iterations. 140
A.3 Occlusion and motion estimates for more frames of the Flower Garden. 141
A.4 Comparison of the occlusion detection accuracy of different variants of proposed technique and two other baseline methods in terms of precision-recall curves and F-measure. . . . . . . . . . . . . . . . . . . 147

A.5 This figure presents the occlusion and motion estimates on the sequences Venus, Grove2 and Grove3 from Middlebury dataset. . . . . 148

A.6 This figure presents the occlusion and motion estimates on the sequences from Middlebury dataset. . . . . . . . . . . . . . . . . . . 149

A.7 Occlusion and flow estimates on the several sequences. . . . . . . . 150
LIST OF TABLES

3.1 Classification results achieved for each combination of dictionary construction method and inference technique. 31
3.2 Confusion Matrices for FutureLight. 33
3.3 Confusion Matrices for Weizmann. 34
4.1 Comparison with other methods. 48
4.2 Representational power of our model as evaluated on the FutureLight dataset. 49
4.3 Confusion Matrix of Future Light Dataset. Overall mean performance 83.87%. 53
4.4 Comparison of classification results. 54
5.1 Comparison of different components of the proposed method. 79
5.2 Experimental results. L is the number of gesture recordings per class. 80
6.1 Performance comparison on KTH dataset. 97
6.2 Performance comparison on ADL. Despite not using structural information or background subtraction, we improve the state of the art by a large margin. 98
6.3 Performance comparison on HOHA Dataset of Individual components of Descriptors. 100
6.4 Performance comparison on HOHA Dataset. 100
7.1 Performance comparison on HOHA Dataset, using different components of our model. 116
7.2 Performance comparison on HOHA Dataset. ............................ 116

A.1 A comparison of the F-measure of our algorithm and [KZ01] on the Middlebury dataset. ................................................................. 143

A.2 The comparison of convergence time of the split-Bregman method and Nesterov’s algorithm. ................................................................. 144

A.3 Quantitative comparison of the proposed models and other robust flow estimation methods in terms of Average Angular Error / Average End Point Error. ................................................................. 144

A.4 Quantitative evaluation of the proposed models on Middlebury test set in terms of Average Angular Error / Average End Point Error. .... 145
I would like to express my deepest gratitude to all those who made this dissertation possible. First of all, I would like to thank my advisor Professor Stefano Soatto for his constant support and encouragement during the course of my Ph.D. His rigorousness and methodical approach to problem formalization and solving have been a source of great inspiration for my research work. In addition, I would like to thank him for his generous feedback about the different facets of my academic life which have been extremely invaluable and insightful. I would also like to thank the my committee members: Professors Alan Yuille, Demetri Terzopoulos, Paulo Tabuada and the former member Professor Petros Faloutsos, for their support and suggestions.

I would like to thank all the members of UCLA Vision Lab who made my stay at UCLA extremely pleasant and delightful. Firstly, I would like to thank the people that they supported me with their friendship and their advise during my first and probably the most difficult year of my doctoral studies: Alessandro Bissacco, Daniele Fontanelli, Gregorio Guidi, Byung-Woo Hong, Haibin Ling, Ryuzo Okada, Jason Meltzer, Eagle Jones. Special thanks goes to Andrea Vedaldi from whom I have learnt many things. His advices have definitely improved me as a scientist and a person. Moreover, I am grateful to Kamil Knuk, Brian Fulkerson, Alper Ayvaci, Teresa Ko, Zhao Yi, Taehee Lee, Yifei Lou with whom I shared most years of my doctoral studies. I would like also to thank Ganesh Sundaramoorthi and Avinash Ravichandaran for providing me priceless feedback during my last years of studies and writing this thesis. Special thank goes to Alper Ayvaci, our collaboration was not only successful but also refreshing.

Furthermore, I would like to express my gratitude to Darko Kirovski and Hugues Hoppe for their support during my stay in Microsoft Research at Redmond. I have learnt many things from them and their advices.
I would especially like to thank my collaborators and friends, Iasonas Kokkinos and George Papandreou, for continuously supporting and wisely advising me. Their dedication and enthusiasm towards research has greatly inspired me from the first time that I met them in the CVSP Lab at NTUA during my undergraduate studies. Furthermore, I express my gratitude to Professor Petros Maragos, my undergraduate advisor, whose inspiring classes and support were fundamental for me coming to UCLA and following my current research field.

Outside the work place, I have been very fortunate to have a group of very supportive and fun loving friends. I would like to thank Ioannis Pefkianakis, Kostas Pelechrinis, Panagiotis Papadimitriou, Christos Boutsidis, Michalis Tzoufras, Christos Vezyrtzis, Nassos Katsamanis, Dimitris Vatakis for all the conversations and happy moments we shared in United States.

Last, but certainly not the least, I would like to thank my parents and my sister. Their unconditional support and their patience were always the strongest source of my motivation and made this incredible journey possible.
1983 Born in Athens, Greece.

2001-2006 D. Eng., Electrical and Computer Engineering, National Technical University of Athens, Greece.

2006-2008 M. Sc., Computer Science, University of California at Los Angeles, USA.

2008 Teaching Assistant, University of California at Los Angeles, USA.

2010 Research Intern, Microsoft Research Redmond

PUBLICATIONS


The analysis of human activities has become a core interest domain in recent years, with a wide range of applications including video indexing, human-machine interfaces and surveillance. Based on the particular application the input source could be obtained from motion capture systems, depth cameras or video cameras. Regardless of the input source, the key challenge in human action recognition lies in explicitly or implicitly modeling the temporal evolution of the data.

Motivated by this challenge, in this thesis we develop techniques for human action recognition and focus on analyzing the spatio-temporal structure of the data. We examine the decomposition of human action to elementary motions and exploit the discriminative power of inference techniques that account for the temporal ordering of the data. Next, we show that capturing the temporal evolution of the human motion explicitly through dynamical systems leads to an accurate unsupervised segmentation of different actions. However, the above methods work best for constrained scenarios such as static cameras. Consequently, for the case of video sequences in uncontrolled environments, we introduce spatio-temporal features that encode the temporal evolu-
tion of local events. Based on these features, we construct a middle-level representation of the video and present a structured action classification model.
CHAPTER 1

Introduction

The human perceptual system has the incredible capability to seamlessly and quickly process visual data thereby interpreting and recognizing thousands of objects in our environment. Humans can not only detect static objects, but also distinguish different types of motion patterns, analyze interaction of objects and recognize complex temporal events. We can immediately determine whether a person is walking, running or dancing even in the presence of a cluttered background, occlusions, and illumination changes. Johansson’s [Joh73] experiments illustrated the ability of humans to reconstruct and recognize complex motion, such as a person running, even in the presence of minimal appearance stimuli. However, understanding complex events using only a single snapshot of the environment can be impossible even for the human visual perception system. Temporal evolution contains a significant amount of information enabling us to make decisions regarding the visual scene.

In order to improve our quality of life, we are interested in enabling machines to interpret the behavior of the various entities in the environment. Such an ability will be useful in several applications such as autonomous cars which can potentially decrease fatal car accidents, intelligent surveillance systems that can identify abnormal events and intuitive computer interfaces that respond to our gestures. To attain these goals, systems need to monitor and understand the environment around them using visual inputs such as images and videos, sensory inputs such as accelerometer data, or more recently infrared and depth cameras. Computer vision has been pri-
arily involved in processing sequential data in order to be able to build such systems. Several algorithms have been developed for addressing different tasks in the various modalities of input signals that are available. For instance, for the case of images, a significant amount of work has been done for tasks such as boundary detection [KWT88, MFM04], segmentation [MS89, SM00] and object categorization [PHS07]. As for video sequences, techniques have been developed for computing the motion between consecutive frames, tracking point correspondences for obtaining an object’s trajectory [SY02, LK81, KWT88, MB04] or detecting and localizing articulated objects [SBR04, VJS05]. In recent years algorithms for “continuous” data modalities such as video sequences have seen an increasing amount of attention due to the advantages such as accumulating and propagating information in the temporal domain, providing a rich source of cues for a variety of low and high level tasks.

In 2010 alone, more than 13 million hours of video were uploaded to Youtube and it is estimated that more than 35 hours of video are uploaded to this service every minute. The constant emergence of smaller and cheaper consumer hardware, such as laptops, mobile phones and high definition video cameras allows one to capture temporal events and to go beyond images of static scenes. Cisco forecasts that 90% of consumer IP traffic will be video data in 2013. This dramatic increase of video data creates the need to automatically analyze the events occurring in video.

The primary focus of this thesis is the analysis of human activities portrayed in video sequences and motion capture data. The analysis of such temporal events can aid in a wide range of applications including human-machine interfaces, surveillance, video indexing, rehabilitation medicine and entertainment industry. However, recognizing the behavior of a person in a video is challenging due to the large variability that exists both in the imaging conditions as well as the way different people perform

\(^1\)http://www.youtube.com/t/press_statistics
a particular action. For instance, different people performing a “simple” action such as walking have different walking pace, style of motion, etc. Moreover, the background clutter that usually exists in video sequences makes the problem of extracting information of a human’s behavior rather difficult. Even if one could extract such information, for instance using motion capture data, where the exact 3-D position of human joints is available across a period of time, understanding the action performed is a non-trivial task. In addition to the large variation of motion data, the high dimensionality is also a significant challenge for the classification problem. The recurring theme of this thesis will be: a) the development and extraction of low level features that capture discriminative statistics, b) the development of discriminative learning algorithms, for recognizing as well as localizing human action in videos.

1.1 Problem Statement and Challenges

Human action recognition is the problem of identifying an action performed by humans from visual data. More specifically, given a video sequence, the goal is to deter-
mine which of a set of predefined action classes best explains the sequence. For example, we want to associate with video sequences extracted from movies, as the frames illustrated in Fig. 1.1, action labels that we have seen before, e.g. “hugging a person”, “running”, “standing up”, etc. As mentioned above, the task of action recognition can be particularly difficult due to potentially large variability in the time dimension, in addition to the difficulties associated with processing of images. However, in many occasions, in addition to addressing the problem of recognition, it is also desirable to know where (both spatially and temporally) an instance of a specific action occurred. This task is called localization.

The words “activities” or “actions” should be used in quotes, since we do not have a precise (operational) definition for them. It is rather difficult to create a valid taxonomy of actions. However, we postulate that such complex phenomena can be understood as the composition of relatively simple spatio-temporal statistics, which we will attempt to characterize. A simple, yet representative example is the motion of the legs during walking. This motion can be decomposed into two components: one when the foot is in contact with the ground and the other when it is swinging above the ground [McG90].

Based on the application, the source of the human action data can be motion capture systems, depth cameras or video cameras. In the case of the common motion capture systems, the exact 3-D location of human joints can be obtained using a set of markers attached to the human body in combination with a rig of infrared cameras. The advantage of using such systems is that the spatial localization problem is not an issue as we know the exact spatial position of human joints at all time instances. Consequently, we can focus our attention on the problem of recognizing the human activity, which is still a challenging problem. We need to develop representations a) that can capture and model the temporal evolution of such data (e.g. temporal evolution of each joint as well as their global configuration), b) that allow for the variations across
different subjects performing the same action, c) that can lead to temporal segmentation of the data by identifying transitions from one action to the other and d) that can be compared using appropriate machine learning techniques for classification.

In the case of video data for applications such as video surveillance, the position of the camera and the environmental conditions (e.g. indoors, background structures) are either fixed or predictable. As a consequence, the task of extracting information relevant to the shape of human silhouette can be tackled using techniques of background subtraction [EHD00, KSE08] and articulated object pose estimation [SBR04, RFZ07]. Further, the analysis of these shape masks has proven promising for recognizing the human action [ZI06, BGS05]. However, in the presence of nuisance factors such as severe scale changes or drastic illumination changes as in an outdoor setting, the extraction of the silhouette becomes very challenging. This can be attributed to the fact that under these settings background subtraction is non trivial. Therefore, in order to have an activity recognition algorithm that it is applicable in realistic scenarios we need to consider features that are robust to noisy shape measurements and account for the spatial and temporal scale variations.

The ultimate goal of computer vision algorithms is to be able to deal with generic image and video data captured under various conditions. Therefore, we also focus on algorithms that successfully explore video data recorded under uncontrolled conditions such as clips recorded with hand-held cameras, cinema movies, broadcasts of sports events. The viewpoint changes, illumination variations, occlusions of humans body parts, background clutter and severe camera motion are some of the main challenges in designing action recognition algorithms. Hence, we seek to develop algorithms that build on local representations that aggregate information through time and they are robust to viewpoint and illuminations changes. Moreover, we need to identify the salient local regions that are correlated with the performed human action and to model
their relative spatio-temporal configuration.

1.2 Thesis Outline

A large part of this thesis explores methods to a) explicitly capture the temporal evolution of the appearance and motion of salient low-level features of the data and b) build mid-level representations that identify the spatio-temporal structure of the performed human actions. The thesis is organized as follows:

In Chapter 2, we review existing methods for action recognition. We will classify these methods into three main classes. This categorization of the methods is mainly based on their data representations. The chapter will also cover common feature descriptors used for capturing local statistics of the video sequence. Moreover, we will also describe classification techniques that use those local representations for action recognition. We will discuss the performance of such methods on standard benchmark databases.

Chapters 3, 4 and 5 introduce algorithms that capture the temporal statistics of human motion based on the assumption that a multivariate time series has been inferred from the person’s motion. Such a representation can be obtained from video, for example by using the silhouette of the person, joint angles of a skeletal model (motion capture data), or sensors worn by an individual (accelerometers). Having identified the importance of features that track the evolution of the action, Chapter 6 and 7 extend the algorithms developed in the previous chapters and propose low-level features based on tracking salient local regions and a middle level representation that models the spatio-temporal structure for action recognition in a video sequence. In chapters 6 and 7, no assumption about the background structure, the illumination conditions, the position, and motion of the camera is made. More specifically:
In Chapter 3, we introduce an approach to action classification which constructs a vector quantization of primitive motions from time series data corresponding to relative limb position estimates. The temporal scale, mean, and shape of primitive motion trajectories are independently modeled, thus creating a flexible dictionary of action primitives. We then explore two inference techniques that leverage our action dictionary representation, and evaluate their performance on both motion capture and video benchmark data.

Chapter 4 investigates dynamical models of human motion that can support both reconstruction and analysis of the data. Unlike coarser discriminative models such as the one proposed in the previous chapter, this model has a fine-scale representational power and can therefore model subtle differences in the way an action is performed. The observed action is modeled as an (unknown) linear time-invariant dynamical system of relatively small order, driven by a sparse bounded input signal. This model can compare the temporal statistics of the inferred input sequences in order to recognize actions and temporally segment the data to individual actions.

In Chapter 5, we present a computationally efficient action classification system for skeletal wireframe motion extracted using a real-time depth sensor. Its key components include an angular representation of the skeleton designed for recognition robustness under noisy input, and a cascaded correlation-based classifier for multivariate time-series data. The classifier is conceived based on the assumption that the input motion adheres to a known, canonical time-base: a musical beat. Consequently, instead of trying to identify the temporal segments corresponding to primitive actions of the actor as in our previous approaches, we correlate our measurements with the time-base of the music, gaining efficiency. The chapter concludes with experimental validation of the proposed algorithm on a benchmark of 28 gesture classes, comprised of hundreds of action instances recorded using the XBOX Kinect platform.
In Chapter 6, we present spatio-temporal feature descriptors that can be computed from video and used as building blocks for action recognition systems. They capture the evolution of “elementary action elements” under a set of assumptions and are designed to be insensitive to nuisance variability (absolute position, contrast), while retaining discriminative statistics due to the fine-scale motion and the local shape in compact regions of the image.

In Chapter 7, we introduce a mid-level approach for action recognition built on top of the feature descriptors introduced in Chapter 6. From each video sequence, we extract salient spatio-temporal structures by forming clusters of trajectories, which serve as candidates for the parts of an action. The optimal assembly of these clusters into a model for an action class is dictated by a graphical model that incorporates appearance and motion constraints for the individual parts and pairwise constraints for the spatio-temporal dependencies among them. During training, we estimate the model parameters discriminatively using SVMs for ranking. During classification, we efficiently fit the model to a video using discrete optimization.

In the appendix, we introduce our contribution to the important problems of dense motion estimation and occlusion detection that are a crucial part of the video feature descriptors introduced in Chapter 6 and 7. Under assumptions of Lambertian reflection and static illumination, the task can be posed as a variational optimization problem, and its solution approximated using convex minimization. We describe efficient numerical schemes that reach the global optimum of the relaxed cost functional. The proposed algorithm is tested on benchmark datasets expanded to enable evaluation of occlusion detection performance in addition to optical flow.
CHAPTER 2

Background

In this chapter, we review existing work in the area of action recognition. Existing approaches in action and activity recognition can be coarsely separated into the following three main classes:

- **Holistic representations** extract global statistics from the entire video volume centered around a person performing an action (given the localization of the person), and compare them using standard norms, block correction, or dynamic time warping.

- **Local feature based methods** describe the entire video sequence as a collection of descriptors of salient regions, without taking into account their spatio-temporal location or their “connection” with the person of interest.

- **Part based representations** attempt to model an action into a middle level representation designed to capture aspects of the local spatial or temporal structure in the data.

2.1 **Holistic representations**

Holistic approaches compute features and develop appropriate metrics using the structure of the entire human body and its temporal evolution for recognizing the performed action. However, the extraction of the shape of the body and its dynamics is based on
“pre-processing” algorithms such as background subtraction [EHD00, KSE08], person detection [DT05, FGM10, DBP10] or tracking [KWT88, SJY06]. The ability of holistic approaches to extract relevant features strongly depends on the success of the “pre-processing” algorithms. Therefore, these approaches are in general successful whenever the video sequence is taken under a static background or the structure of the scene makes human person detection and tracking feasible.

Most holistic methods use the silhouette of the person to represent the action, whereas others use the dense optical flow in the region of the person. The advantage of using dense optical flow is that it avoids the artifacts that noisy silhouettes create. Yamato et al. [YOI92] modeled the evolution of the binary silhouette of a person playing tennis using Hidden Markov models (HMMs) [RJ93]. The observations of the HMM were quantized feature vectors capturing at each dimension the ratio of pixels that belong to the human silhouette versus the background at a given partition of the image. Darrel and Pentland [DP93] created a set of view models by interpolating a relatively small set of key frames of the performed gesture, and applied dynamic time warping [SC78] to match the test image’s normalized correlation score against the view models to recognize hand gestures.

Temporal templates that capture the characteristics of the entire space-time action volume and not only key frames [DP93] were introduced by Bobick and Davis [BD01]. Using the difference between successive frames, the authors constructed two 2-D templates: the motion energy image indicating the locations of motion and motion history image encoding with higher intensities the locations where motion was detected more recently. For recognizing the templates, higher-order moments were computed from the motion history and energy images and a Mahalanobis distance was calculated between the input descriptor and each template of the labeled database.

Blank et al. [BGS05] extended a methodology for analyzing 2-D shapes to deal
with space-time action shapes extracted using background subtraction. This method used properties of the solution of the Poisson equation to extract features measuring the space-time saliency, the local orientation, and aspect ratio of different space-time parts. Similarly, Zelnik-Manor and Irani [UI06] characterized the space-time volume of an action by spatio-temporal features at multiple temporal scales to deal with the temporal variations between different action performances. The proposed spatio-temporal features captured the local surface orientation, which is highly correlated with behavior of the moving human.

Efros et al. [EBM03a] proposed an action recognition method based on low level features of the optical flow field on the local neighborhood of the human of interest. Optical flow features are less sensitive to cluttered background than shape-silhouette based features. Moreover, optical flow features provide information about the motion of the entire human figure and not only its boundary. However, the computation of the optical flow between successive frames is an ill-posed problem (see Appendix) and existing approaches [BBP04, WPZ08, LK81] lead to noisy estimation. Therefore, Efros et al. smoothed and spatially aggregated optical flow data to form a more stable frame-wise spatio-temporal motion descriptor. For classification each frame’s motion descriptor was labeled independently by retrieving neighbors from a database of annotated video sequences. Fathi and Mori [FM08] used the same human-centric representation in combination with Adaboost [FS95] to provide a more robust action classification system. However, the appearance of the human encapsulates a significant amount of cues that optical flow based features fail to encode.

Ke et al. [KSH07] proposed a method based on hybrid features of shape and optical flow that correlates spatio-temporal volumes to video sequences, without requiring background subtraction. The test video was segmented in space-time using mean shift [CM02]. A matching algorithm was employed to calculate the distance between shape
and optical flow features of the action template and the over-segmented video. The downside of template based matching algorithms is that they are not invariant to nuisance variations such as vantage point, scale, or partial occlusions.

### 2.2 Local feature based methods

Methods based on local features decompose the video into small spatio-temporal regions and capture their shape and motion characteristics. In contrast to holistic representations (Section 2.1), the local spatio-temporal regions are not human centric. In such methods, all the location information about the features is ignored. Local feature methods have been shown to be very effective in the problem of object category recognition [CDD04] from single images and their extension in the spatio-temporal space has also shown promise for action analysis tasks. The extracted set of features are invariant (or at least insensitive) to transformations of the image. Using these features leads to the elimination of some nuisance factors (such as small changes in viewpoint or lighting).

The pipeline of local feature methods can be divided into three parts: 1) feature detector, 2) feature descriptor and 3) representation of the video. The feature detector seeks to find points of the video or image that may be reliably found in another video or image. They usually select those points as the local maximum of image functions, e.g. the determinant of the structure tensor. Some commonly used feature detectors in images are the Maximum Stable Extremal Regions [MCU04], Scale Invariant Feature Transform (SIFT) [Low99, Low04] and the Harris corner [HS88]. The latter two were successfully extended into video sequences by the work of Scovanner et al. [SAS07] and Laptev and Lindeberg [LL03] respectively. Feature descriptors capture the local statistics of small image regions such that they are invariant to transformation of the image. The final representation that can be used in combination with local features may
vary. One of the simpler and relatively effective representations is the *bag-of-features* approach [CDD04]. The *bag-of-features* approach represents the entire video as an unordered set of descriptors. All spatial and temporal information is discarded, like as the descriptors have been placed in a “bag” without any order information.

Laptev and Lindeberg [LL03] proposed an interest point detector that evaluates the cornerness criterion by extending the work of Harris and Stephens [HS88]. Local maxima of a function that depends on the eigenvalues of the 3-D structure tensor of the video, defined the salient points of the sequence. Laptev and Lindeberg [LL04] used high-order derivatives (spatio-temporal jets) computed on a cube centered to each detected salient point to capture the information about the local motion and the local spatial appearance. Laptev *et al.* [LMS08] provided a more robust description of the local spatio-temporal neighborhood of the salient point by using histograms of gradient (HOG) and histograms of optical flow (HOF). The histograms were aggregated in local sub-regions defined by a $N \times N \times M$ grid centered to the salient point.

Dollár *et al.* [DRC05b] introduced a salient point detector that is more sensitive to changes of motion direction compared to the one proposed by Laptev and Lindeberg [LL03]. The detector selects interest points as local maxima of a filtered image. The filter is a combination of a temporal 1-D Gabor filter and a spatial 2-D Gaussian kernel. Similar to Laptev and Lindeberg [LL04], Dollár *et al.* [DRC05b] investigated different choices of spatio-temporal descriptor to capture the local statistics at each detected interest point of the video. They concluded that the descriptor that yields the best performance was the vectorized spatio-temporal gradient of the pixels of the cube. This descriptor retains the positional information in contrast to the histogram based descriptors of Laptev *et al.* [LL04, LMS08]. However, it loses the benefit of increased robustness that histogram representations have, e.g. SIFT [Low99], HOG [DT05].

Kläser *et al.* [KMS08] presented a descriptor for spatio-temporal cubes that is
based on histograms of 3-D gradient orientations. This descriptor can be characterized as an extension of the SIFT descriptor [Low99, Low04] in the spatio-temporal domain, similar to the work of Sconvanner et al. [SAS07]. The main difference between the two works is the way that the gradient orientations are quantized. [KMS08] used a regular polyhedrons that avoids singularities that appear if orientations are expressed in polar coordinates [SAS07].

The blob detector [Bea78] based on the determinant of the Hessian of an image was extended to spatio-temporal cases by Willems et al. [WTV08]. The detector is computationally efficient and provides a denser coverage of the video with salient point candidates. Willems et al. [WTV08] extended the SURF descriptor [BTV06] to videos. The spatio-temporal cube was uniformly split to $N \times N \times M$ cells. Each cell was represented by a vector of weighted sums of uniformly sampled responses of Haar-wavelets along the three axes. A more detailed evaluation of the above local spatio-temporal features can be found at [WUK09].

Recently, local methods have been developed that capture the characteristics of spatial interest points tracked over time, instead of describing a local neighborhood around a salient point [LL04, LMS08, KMS08, DRC05b, WTV08, SAS07, NBT07a]. The trajectories of the interest points provide the means for developing long term analysis of the evolution of both the local motion and appearance. The long term analysis enables to link together cues of the performed human action and it is a key component of all the algorithms presented in this thesis. Moreover, the use of trajectories guarantees that the actual feature descriptor will be directly linked with a particular moving image salient feature. In contrast, cube representations blend sources of information that are not directly correlated with the detected interest point.

Sun et al. [SWY09] detected spatial salient points employing SIFT detector and tracked them over time by establishing correspondences in a local window. The ap-
pearance of the local patch tracked was described by the average SIFT descriptor. In addition, the stationary statistics of the Markov chain of instantaneous velocities described the evolution of the trajectory, which, however, suffers from small-sample effects and does not provide information for the motion of the local neighborhood around the trajectory. Similar low level feature descriptors were employed by [KB10].

Matikainen et al. [MHS09, MHS10] extracted trajectories using the KLT tracker [LK81]. The extracted trajectories were forced to a fixed length, so that the clustering can be performed using K-means [Bis06]. In order to also capture the information of the motion of neighbor trajectories, the authors performed clustering that account for possible affine transformation between the cluster center and the actual trajectory. As a consequence, the trajectories that lie on the same rigid object were assigned to the same cluster center. The final trajectory descriptor was the elements of an affine transformation matrix that characterizes the corresponding cluster center concatenated with the velocity vectors of the cluster center trajectory. However, this descriptor was only able to capture information about the local motion statistics and neglected the local appearance of the tracked region. Moreover, enforcing fixed length trajectories lacks the ability to discriminate between events that are trackable over a long period of time and others due to their own characteristics of their motion, e.g. fast motion, self-occlusions, etc. Messing et al. [MPK09] also used the standard KLT tracker [LK81, Bir96] and quantized the varying length trajectories as time series of log-polar quantized velocities. These velocities were then modeled as observations in a sequential graphical model.

Most of the local feature methods [LL04, KMS08, DRC05b, WTV08, SAS07, KB10, MHS09, MHS10] employed a bag-of-features representation, discarding much of the spatial structure of the data. The main assumption of this representation is that the video can be associated with a distribution of local features. In order to deal with
the dimensionality of the local feature descriptors a quantization step is applied. Then a dictionary is formed from the training data and each extracted descriptor is assigned to a dictionary element. The formation of the dictionary is a crucial step for the discriminative ability of the representation and the effectiveness of the final classifier. The definition of the appropriate distance measure in the feature space is important, and not trivial in the case of varying length descriptors as the ones associated to trajectories (see Chapter 6).

The understanding of the spatial relationships in image or video reveals important information about the objects in the scene and their activities. A number of algorithms have been proposed in object recognition literature based on the bag-of-features framework to capture local spatial characteristics, e.g. spatial pyramid matching [LSP06]. Laptev et al. [LMS08] extended the work of Lazebnik et al. [LSP06] and showed that encoding the spatio-temporal layout of a video using a fixed space-time grid improves the recognition performance compared to bag-of-features approaches [SLC04]. To maximize the recognition accuracy, Sun et al. [SWY09] adapted Multi-Kernel methods to learn the optimal weights between the several different feature channels obtained from the fixed space-time grid. These methods, though, again encode the video as a collection of statistics without any notion of selection of regions that are relevant to a particular human motion. In contrast, the goal of part based representations (Section 2.3) is to identify and model only the regions correlated to a particular action.

2.3 Part based representations

The third class of approaches attempts to decompose an action or activity into “parts” that are designed to capture aspects of the local spatial or temporal structure in the data. Sequential data models have been employed to represent the temporal variabil-
ity [WB99, SFP00a, IF07, POW10], or the entire time series is retained. For instance Brendel and Todorovic [BT10] used a time series of activity codewords, identifying at each frame only one promising region as a part of an activity and enforcing the temporal consistency with a Markov chain. The authors segmented each video frame using mean-shift algorithm [CM02]. The segmented regions were then described by spatio-temporal HOG [DT05, KMS08]. Given a video, a time series was extracted by selecting a single region that best match with the discriminative learned activity codewords at each frame. Brendel and Todorovic [BT10] employed a standard Viterbi algorithm to optimize the previous selection process in order to optimally fulfill spatial smoothness and temporally consistency constraints. This approach, however, makes the strong assumption that the range of the performed action spans the whole duration of the video. Moreover, the fact that the action model cannot capture spatio-temporal relationships between different salient regions of the video suppresses the discriminative power of the algorithm.

To address this problem Wang and Mori [WM10] proposed a more complex part-based model that encode spatial pairwise relationships. More specifically, Wang and Mori [WM10] introduced a model based on the hidden conditional random field (HCRF) [QWM07] for object recognition, whereby pairwise relationships among spatial image patches are encoded explicitly, but classification was performed on a frame-by-frame basis considering only the spatial grouping. The model combined global features (similar to bag-of-features representation of the video) and local patch features, that localized the salient regions of the image where the action was performed.

Niebles et al. [NC10] extended the notion of a part from a spatial segment [BT10, WM10] to a set of consecutive video frames. The proposed model, similar to [WM10], was composed of global and local features. The discriminative model formulation is an extension of the very successful part based model of Felzenszwalb et al. [FGM10] for
object recognition. The local features decompose the video to a set of shorter temporal segments, each with its individual temporal range. This enables temporal composition. However, the ensuing model lacks the ability to spatially localize action parts, since each video segment is represented as a collection of spatio-temporal interest points [SLC04].

The approach described in Chapter 7 falls in the part based representation class. However, we extend the methods of [NC10, SFP00a, WM10, DRF10] a step further and encode both the spatial and temporal structure of the action, enabling part localization both in the space and time domain.
CHAPTER 3

Flexible Dictionaries

3.1 Introduction

In this chapter, we focus on classification of events with distinct temporal signatures. In contrast to some recent work that reasons in the spatio-temporal domain (Section 2.2), we do not deal with pixel or feature-based appearance during motions deemed to be interesting [NWF06, NBT07b, DRC05a]. Instead we focus on the source of such appearance variations: the deformation of the human body. However, we do not seek to explicitly identify dynamical systems that drive a particular human skeleton [BCM01]. Instead we seek to learn primitives of the motion of human limbs through examples, and perform classification decisions using a dictionary-based representation.

In an attempt to capture the available temporal information, we represent actions as multi-dimensional time series. We capitalize on this representation to easily extract trajectories of simple body movements. By clustering such “primitive” trajectories, we seek to construct a nonparametric quantization of the physically possible motions, as they appear in the actions that we are interested in classifying. Because limb motions contain a large amount of variability in temporal scale (speed of the action), as well as their relative location with respect to the center of the body, we explicitly and independently quantize these informative components to more effectively represent and match primitive trajectories.
We pursue the above goals by constructing dictionaries of action primitives. The dictionaries are obtained by clustering windows extracted at interesting points in our time series representation of actions. We demonstrate the performance of our dictionaries by applying two types of inference techniques. Our results show that:

1. Both of our inference techniques discriminate actions with well above 90% accuracy on commonly used benchmark datasets, using only a simple low-dimensional parameterization of human motion that we can track over time in a fully automatic fashion.

2. When exact pose is unavailable, our approach is robust to using a very rough approximation (bounding box), allowing us to succeed where other pose-based classifiers require supervision in the pose tracking procedure.

3. When exact pose information is available, such as in motion capture data, we show that we outperform recent dynamics-based approaches by up to 8.33% mean classification accuracy.

Our work attempts to capitalize on the best of local feature representations (Section 2.2) and model-based approaches. The model-based approaches characterize the human motion by the parameters of a model that is fit to the data. Hidden Markov Models [WB99, SFP00b] and finite-state models [IF07, HNB04] have been used to model the temporal variability of human motion. However, these methods lose the details necessary to infer the dynamics as a result of the coarseness of their representation. Others model the dynamics of human gaits using hybrid linear models [PRM00, VSS03, AT04, BS06, LTS07], which exhibit great generative power; however, their discriminative power decreases with increased model complexity. This tradeoff makes it difficult to classify actions such as dancing. Other model-based approaches propose to design invariants of the motion sequence [SAM07]. However,
these methods are difficult to generalize and require many samples of each action.

With our flexible dictionaries of action primitives we create a nonparametric model of possible human motions. The unique element of our approach is that such a model allows us to use statistical inference techniques typically reserved for holistic action approaches.

3.2 Actions as Time Series

As noted in Section 3.1, in this chapter we take a less common view of actions. Instead of representing an action as a collection of spatio-temporal patches [NWF06, NBT07b, DRC05a], we interpret human motion as a multi-dimensional time series that captures the deformations of the body during activity. In particular, we track the positions of limb endpoints relative to the center of the body. As in [SAM07], these include the arms, legs, and head (only 5 points).

In the case of motion capture data, where positions are expressed in three dimensions, our chosen representation produces a 15 dimensional time series \((x, y, z)\) for each limb endpoint). For video we do not attempt to estimate 3D pose. Instead we roughly capture body deformations in the image plane. Because extraction of time series from video can be dataset dependent, details are provided below in the experimental section.

It is important to note that our methods are not bound to the above representation and can be used with time series generated from video or motion capture data using any number of techniques.
3.3 Dictionary of Action Primitives

Throughout this chapter we define an action primitive as a time series subsequence that encodes a single dimension of a commonly occurring deformation. Due to our chosen method for obtaining time series, in our case such a primitive actually corresponds to a projection of a simple limb trajectory onto a single axis. For example, for a 3D trajectory of a single limb the time series resulting from the $x$, $y$ and $z$ coordinates each have their own action primitives. Independent action primitive dictionaries are constructed for all dimensions. This, for example, means 15 dictionaries in the case of motion capture data.

In the following sections we explore three incrementally more complex dictionary building procedures and their associated strengths and weaknesses for particular tasks. Each successive technique incorporates an additional degree of invariance by explicitly modeling a particular informative characteristic of action primitives.

3.3.1 Construction of a Simple Dictionary

The most basic approach to dictionary construction is to simply extract and cluster time series sub-sequences of a fixed size. Such clustering must be done with care, as recent literature on time series classification has shown that failing to tailor the sub-sequence sampling procedure to the signal can introduce artifacts that overpower the input, producing sinusoids independent of the underlying data [LKT03, FHN08]. For this reason we sample the signal by extracting sub-windows only around extrema points, which achieves local translation invariance as illustrated in [Soa07]. To counter the effects of noise when detecting extrema, a small initial smoothing is applied. Sampling at the extrema of all the time series captures all significant changes of motion direction within each dimension of an action.
Once sub-windows are extracted for all dimensions in all action samples, we quan-
tize them independently for each dimension using $K$-means. The $\ell_1$ distance was used to compare sub-windows. As shown in Fig. 3.1a, the representative power of this simple approach suffers because several dictionary elements are wasted on encoding the same primitive shape occurring at different amplitudes. Thus, in our next approach we seek to increase representative power by independently quantizing the mean of the sub-windows.

### 3.3.2 Increasing Representative Power by Mean Quantization

To achieve clustering of trajectories invariant to their average amplitude, we subtract the mean from all extracted sub-windows before quantizing them as above. The average amplitude of a primitive trajectory is likely to be informative for discriminating actions, as it captures the relative position of the limb during motion. Thus we quantize the subtracted means. Hence, in addition to one dictionary of primitive trajectories per dimension of the time-series, we also add quantization of the mean. Each window extracted from the time series composing an action can now be represented with a best matching canonized trajectory and an assignment to a particular bin of the quantized mean.

Fig. 3.1b shows that this approach generates a dictionary spanning a greater variability of trajectories, thus improving its representational and discriminative power. This dictionary construction approach was found to perform well in classification tasks. However, modeling the temporal scale of dictionary elements still proved essential for achieving best results.
Figure 3.1: The action primitive dictionaries constructed without, Fig. 3.1a, and with, Fig. 3.1b, mean quantization for the $x$-coordinate of the left hand in the Future-Light motion capture dataset. The dictionaries were constructed with $K$-means, using $K = 20$, and a sub-window size of 21. Notice that in Fig. 3.1b several clusters (elements 2, 5, 6, and 7 for example) capture the same trajectory, but at different mean locations. In addition to being inefficient, this means that when we represent a signal with this dictionary, much of the effort will go to matching the mean, diminishing the significance of the trajectory (shape). In Fig. 3.1b, the mean was quantized (lower right) into 5 bins independent of the trajectory quantization. One can see, by comparing the two figures, that once the mean was modeled out from the trajectory quantization much more variability was captured in the 20 trajectory elements.

### 3.3.3 Modeling Time Scale for Temporal Flexibility

Human actions can have large temporal variability. Even within a single action instance the same primitives may be performed at greatly varying speeds. Picking up an object, for example, may be performed very rapidly when the object is solid or very slowly when the object is fragile and caution is required. Therefore, to increase
the accuracy with which the dictionary can capture our data it is important to be able to represent action primitives at different time scales. For this reason, we extend our sub-window extraction procedure.

As before, the centers of time series sub-windows are selected at extrema points. However, instead of using a preset size we automatically select the size of each sub-window. This is done by symmetrically growing window borders until the nearest extremum is encountered.

Once the length of all sub-windows is known, we quantize this temporal scale using $K$-means. Each extracted sub-window is then matched to the closest quantized window size and resampled to this canonical size. Interpolation with B-splines was used to minimize loss during sub-window resampling. By this quantization of the time scale, we avoid comparing large scale phenomena with small scale trajectories, while maintaining invariance within local scale ranges.

This procedure of rescaling should be understood in the context of achieving invariance with respect to re-parameterization of the time axis, as an alternative to performing dynamic time warping when comparing two time series. However, both the extraction of invariants and comparing modulo re-parameterization of the temporal axis should be performed in a way that respects the dynamics of the underlying signal [Soa07].

Following the rescaling, the mean quantization is performed as before, across all the extracted sub-windows. Finally, the clustering of the shape of the trajectories is performed independently within each canonical window size, thus creating a dictionary of action primitives spanning multiple time scales.

Adaptive window size selection does not only offer us a way of modeling the temporal variability, but also narrows down the importance of the choice of the window size, which needed to be manually specified in the previous approaches.
Moreover, as hypothesized, the power of the multi-scale dictionary becomes evident when applied to the reconstruction task, e.g. Fig. 3.2. The dictionary is able to capture the major components of variation of the time series. In order to generate the new canonical signal, a B-spline is fitted to the concatenated best matched dictionary elements.

Figure 3.2: Reconstruction of a time series representing a coordinate of the relative position of one of the limbs using the best matching dictionary elements. The original signal is shown in blue, and the approximation by the dictionary is shown in red. Fig. 3.2a shows a case where the quantization of the time scale, shape, and mean contains the correct elements needed for reconstruction. The example in Fig. 3.2b shows a case where several very rough approximations are made due to the coarseness of our quantization (mean $K = 5$, time scale $K = 5$ and shape at each scale $K = 4$).
3.4 Classification

We show that our dictionary of action primitives representation naturally lends itself to the action classification task, enabling a range of inference approaches. In particular, we demonstrate two inference techniques: A bag-of-words approach, which ignores temporal constraints, and a string classification technique that explicitly takes into account the sequence in which action primitives occur.

3.4.1 Characterizing an Action

Once dictionaries are constructed for all dimensions, any action can be expressed by projecting each component time series onto the appropriate dictionary. The projection procedure follows a similar pipeline to dictionary construction. First, sub-windows are extracted in the same manner as during dictionary construction. Each sub-window is then matched to the closest quantized scale, mean, and trajectory in the given dictionary and assigned unique labels indicating the correspondences. For the simplest dictionaries only the trajectories are matched, and for the mean-quantized dictionaries only mean and trajectory are considered. In the case of multi-scale dictionaries the trajectory is resampled to the closest canonical size before comparing mean and shape. The result of projecting a time series onto a dictionary is thus a sequence of labels annotating all interest points.

3.4.2 Bagging Action Primitives

Our first classification approach discards all causal constraints. It is thus meant to serve as a baseline for more complex techniques. In this approach an action is characterized by comparing the distribution of dictionary elements within the whole action. This is known as the bag-of-words approach. It has shown to be very effective in the domains
of document classification, category recognition in images, as well in behavior analysis [DRC05a, IF07]. To classify actions with this technique, we build a histogram of labels for each dimension of the time series. The dimensionality of the histogram for each time series is equal to the number of elements in the dictionary used. Each histogram bin records the number of times a particular dictionary element occurred during the action. For mean quantized dictionaries, we consider distributions of mean and shape in independent histograms. We found empirically that this independence assumption shows almost no difference from results given when performing inference with multi-dimensional histograms to couple the mean and trajectory labels. Each histogram is turned into a distribution independent of action duration via normalization. Finally, all histograms from each dimension are stacked to create a single action descriptor.

Once the stack of histograms characterizing each action is computed, we use a standard supervised SVM classification approach [Vap95]. Given a partition of the action samples into labeled testing and training data, we train a multi-class $\chi^2$ kernel SVM on the histograms. Any action or action segment can then be classified with the SVM, once it is converted to the histogram of action primitives representation.

### 3.4.3 Exploiting Temporal Constraints

To incorporate temporal constraints into our inference, we map the problem of classifying time series to a problem of classifying strings [TV06, YGS07]. By projecting the interest points of a time series to our dictionary, as described, we obtain a sequence of labels which can be viewed as a “string” representation of the time series. The alphabet elements are the elements of our dictionary. Our approach is motivated by algorithms from the domain of protein sequence similarity detection [LN02]. Our classification procedure can be described in three steps:

First, we compute the pairwise similarity between all the string sequences that de-
scribe each sample of an action. There are several techniques for string matching in
the literature [SW81, NW70], that are commonly used to align proteins or nucleotide
sequences. In our implementation, we used the Smith-Waterman (SW) algorithm; a
dynamic programming algorithm which finds the local alignment of two string se-
quences while allowing gaps. In order for the similarity score to be independent of the
length of the two sequences that we compare, we normalize the score by the length of
the aligned sequences. Similarities computed for each dimension of time series of an
action are equally weighted so as to obtain an overall similarity score for each action
sample. We also experimented with jointly aligning all channels with respect to a sin-
gle temporal warping, as opposed to independently aligning each channel. While this
approach is physically more plausible (there is, after all, one temporal dimension), the
resulting model achieves classification results that are slightly worse than those using
independent alignment on each channel. This empirical finding remains to be fully
understood.

The second step is to represent each action via its similarity to the actions in the
training set. More specifically, the action $F$ is represented by a vector of scores:

$$X_F = [x_{f_1}, x_{f_2}, \ldots, x_{f_n}]$$

(3.1)

where $f_i$ is the $i$th training set sequence, and $x_{f_i}$ is the value of the SW score between
sequence $F$ and $f_i$. This procedure can be viewed as the vectorization step of a kernel
method approach, such as a SVM.

The third step is the definition of the positive definite kernel function:

$$K(X, Y) = \frac{X \cdot Y}{\sqrt{(X \cdot X)(Y \cdot Y)}}.$$  (3.2)

This kernel $K(\cdot, \cdot)$ is then transformed into a radial basis kernel using the induced
distance:

$$D(X, Y) = \sqrt{K(X, X) + K(Y, Y) - 2K(X, Y)}$$  (3.3)
Once we have the radial basis kernel, we train a multi-class SVM, and then classify actions.

### 3.5 Experiments

We applied both of our inference approaches to human motion classification in both motion capture and video data. Both fixed and adaptive window size dictionaries were used. We show that when using motion capture data, the ideal case for our approach, we achieve state of the art performance, demonstrating 5 to 8% improvement in classification over recent dynamics-based approaches. We also show that our methods perform competitively in video, despite the availability of only a rough estimate of relative limb locations.

#### 3.5.1 Datasets

As presented in [SAM07], the FutureLight motion capture dataset [R] contains a total of 158 action samples which represent 5 actions, with a variable number of samples of each. The actions include “Dance”, “Jump”, “Sit”, “Run”, and “Walk”. Each action contains considerable intra-class variability, including several fairly ambiguous samples, such as a ballet dance that appears visually similar to a walk. The durations of captured actions vary from 100 to over 800 frames. As mentioned earlier, we choose to represent each recorded action with a 15 dimensional time series.

To demonstrate the applicability of our approach to video-based action classification, we evaluate performance on the Weizmann Human Action dataset provided by [BGS05]. This dataset contains video of 9 different people, performing 9 unique actions, including: run, walk, sideways run, jump, jump in place, jumping jack, one-handed wave, two-handed wave, and bend. The dataset is annotated with silhouettes.
Table 3.1: Classification results achieved for each combination of dictionary construction method and inference technique.

<table>
<thead>
<tr>
<th>Classification Results</th>
<th>FutureLight</th>
<th>Weizmann</th>
</tr>
</thead>
<tbody>
<tr>
<td>our BAG+SVM</td>
<td>95.30</td>
<td>91.36</td>
</tr>
<tr>
<td>our SW+SVM</td>
<td>96.04</td>
<td>97.52</td>
</tr>
<tr>
<td>our BAG+Temporal Scale+SVM</td>
<td>97.40</td>
<td>95.06</td>
</tr>
<tr>
<td>our SW+Temporal Scale+SVM</td>
<td><strong>98.03</strong></td>
<td><strong>100</strong></td>
</tr>
<tr>
<td>[SAM07]</td>
<td>89.7</td>
<td>92.6</td>
</tr>
<tr>
<td>[BGS05]</td>
<td>–</td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>

for each sample action, obtained using a background subtraction algorithm. Based on the silhouette properties we represent each action as a 5 dimensional time series. Specifically we split the silhouette into 4 quadrants at its center of mass and track the max width of the silhouette in each quadrant. We also track the absolute height of the silhouette over time. These features provide a very rough estimate of the positions of the body extrema over time and it is computationally more efficient than the spatio-temporal “cubes” that [BGS05] used or the orientations of rectangular patches of the silhouette used by [IF07].

3.5.2 Results

Performance was evaluated on the datasets above using a leave-one-out cross validation procedure. Below we report the best results achieved for each combination of dictionary type and inference technique after exploring a range of various parameters: Surprisingly, when ignoring global temporal information and using the bag-of-
words classification approach, we were able to achieve excellent classification performance in both motion capture and video data. Using dictionaries with fixed window size (BAG+SVM) we obtained 95.3% mean classification accuracy on the FutureLight dataset (Table 3.2 (1), window size: 21 samples) and 91.36% mean accuracy on the Weizmann action dataset (Table 3.3 (1), window size: 17 samples). Using multi-scale dictionaries (BAG+Temporal Scale+SVM) we obtained 97.40% mean classification accuracy on the FutureLight dataset (Table 3.2 (3)), and 95.06% mean classification on the Weizmann dataset (Table 3.3 (7)).

Incorporating temporal constraints (SW) with our second inference method, we obtained improved performance in both datasets for fixed window and adaptive window size dictionaries. This is shown in Tables 3.2 (2) and (4) for the FutureLight dataset and Tables 3.3 (2) and (4) for the Weizmann dataset. For the FutureLight dataset performance increased by a little over a half percent, but for classification in the Weizmann dataset, we obtained a significant improvement of roughly 5 to 6%. The small increase in improvement in FutureLight is due to the already excellent performance of the bag-of-words approach. It is also worthy to note that these datasets do not contain actions where the global temporal order of the action is crucial.

3.6 Discussion

In this chapter, we proposed an approach for constructing dictionaries of primitive actions suitable for action classification from time series data. We explored the representative and discriminative power of these dictionaries using them in reconstruction and classification tasks, respectively.

We achieved state of the art classification results in the FutureLight motion capture dataset using both bag-of-words and string matching inference techniques. Also, we
Table 3.2: Confusion Matrices for FutureLight (Tables 1-4)

<table>
<thead>
<tr>
<th></th>
<th>Dance</th>
<th>Jump</th>
<th>Run</th>
<th>Sit</th>
<th>Walk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dance</td>
<td>28</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump</td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Run</td>
<td>1</td>
<td>28</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sit</td>
<td></td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Walk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>48</td>
</tr>
</tbody>
</table>

Table 1. BAG+SVM

<table>
<thead>
<tr>
<th></th>
<th>Dance</th>
<th>Jump</th>
<th>Run</th>
<th>Sit</th>
<th>Walk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dance</td>
<td>29</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump</td>
<td></td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>4</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sit</td>
<td></td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Walk</td>
<td></td>
<td>48</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. SW+SVM

<table>
<thead>
<tr>
<th></th>
<th>Dance</th>
<th>Jump</th>
<th>Run</th>
<th>Sit</th>
<th>Walk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dance</td>
<td>29</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump</td>
<td></td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>4</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sit</td>
<td></td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Walk</td>
<td></td>
<td>48</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. BAG+Temporal Scale+SVM

<table>
<thead>
<tr>
<th></th>
<th>Dance</th>
<th>Jump</th>
<th>Run</th>
<th>Sit</th>
<th>Walk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dance</td>
<td>28</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump</td>
<td></td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>29</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sit</td>
<td></td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Walk</td>
<td></td>
<td>48</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. SW+Temporal Scale+SVM
Table 3.3: Confusion Matrices for Weizmann (Tables 1-4)

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>A8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>A9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1. BAG+SVM

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>A8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>

Table 2. SW+SVM

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>

Table 3. BAG+Temporal Scale+SVM

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>A8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>

Table 4. SW+Temporal Scale+SVM

demonstrated competitive results in video data, while using significantly simpler features with only 5 dimensions. Together, these results make a compelling argument for viewing actions from the perspective of multidimensional time series.
CHAPTER 4

Spike Train Driven Dynamical Models
for Human Actions

4.1 Introduction

For some tasks, such as classification of distinctive motions, purely discriminative models are sufficient [EBM03b]. Some benchmark datasets can even be classified reliably by taking into account as little information as local shape and optical flow in a single frame [SV08]. However, in situations where the temporal order of motions is significant (or perhaps the only discriminative information), or to address more subtle queries such as long-term or fine scale prediction, models with generative capability and greater representational accuracy are useful. Since the discrete multinomial state of generative models, such as Hidden Markov Models (HMMs) [FP02, WB99, IF08], experience an exponential increase in parameters as more signal history is encoded, we favor dynamic models with continuous latent variables to pursue the desired level of detail in action representation.

In this chapter, we propose to view human motion analysis as a blind system identification where each limb is an unknown linear dynamical system (LDS) driven by an unknown input. Intuitively, the dynamical model represents physical characteristics of an actor, such as mass and inertia, whereas the input represents the driving signal, a signature of the action. Without additional constraints this is an ill-posed problem.
Traditionally, assumptions have been made that the driving input is a process with samples from some canonical distribution (typically a Gaussian). These approaches were successful at capturing observations with second-order stationary statistics, and therefore worked well for modeling quasi-repetitive actions such as walking and running [BCS07]. However, the limitations of these models become quickly apparent when one considers more complex non-stationary sequences, e.g. Fig. 4.1. Our goal in this work is to be able to capture such non-stationarities in human action sequences and to reliably identify when changes between distinct actions occur.

To render the blind identification problem above well-posed we constrain the dynamics to be linear and time-invariant (our body masses do not change at the time-scale of observation), transferring all the non-stationary characteristics of the observed time series to the input. Ideally, we want a class of inputs that would serve as a signature for actions. One logical option is to assume that the input should be mostly zero, except when soliciting a change of elementary movement. When non-zero, it should have bounded energy, to avoid embarrassing violations of elementary physics laws. This translates into the problem of performing blind identification/deconvolution under bounded energy and sparsity constraints on the input. Exploiting recent results from convex optimization and sparse representations, in Section 4.2 through Section 4.4 we develop a new algorithm for the task.

We validate our model by demonstrating the ability to accurately capture more complex actions than previous linear dynamical system approaches in Section 4.5. The compressive power of our sparse representation is also addressed. In Section 4.6, we present the application of customizable synthesis by modification of model parameters and show that our sparse representation supports segmentation and classification tasks. Through the segmentation and classification tasks, we confirm our intuitions that the input encodes signatures of actions.
4.1.1 Related Work

The model proposed in this chapter falls into the class of linear dynamical systems, where the task of motion modeling has been posed as a system identification problem [BS06, PR00]. Up until now the LDS literature in human motion has assumed a stochastic input with a known distribution, which limits the representational capability to simpler regular actions. This motivates the use of switched-linear dynamical systems (SLDS), in which changes of the model parameters enhance the ability of the model to capture more complex motions [PRM00, ORB05]. In [PFH99], an SLDS approach was proposed where only the zeros of the transfer function were allowed to change across actions and an HMM was used to drive these changes. Works with a similar spirit have used switched autoregressive (SAR) systems to model videos. Video segmentation is achieved by detecting changes of the coefficients of the AR model. The identification of SAR has been addressed as a convex optimization problem by [OSC08], and as identification of homogeneous polynomials by [Vid08].

Yet another perspective on capturing the non-stationarity of human actions are Gaussian processes [WFH08]. These models learn a nonlinear mapping from the observation space into a latent space and a nonlinear system in the latent space. A downside of this approach is that it does not provide information which can directly be used for classification or segmentation of the modeled motion. Physically based nonlinear temporal models have also been used to synthesize human motion [FPT01, HW95]. However, the process of concatenating “basic” controllers becomes too complex for most actions of interest.

In our case we assume a single linear time-invariant model. We show that by changing the assumptions on the input we increase the ability of LDS to capture complex actions and simultaneously capture useful action characteristics in the input.

In this chapter, like most of the literature above, we focus on designing dynam
ical models for actions, with no regard to how the time series is extracted. In our experiments, we use motion capture data to evaluate our approach.

4.2 The Underlying Linear Dynamical System

Data observed from human actions, whether from video or motion capture, can be viewed as a multivariate time series. The core hypothesis of the work in this chapter is that such multivariate time series $y(t) \in \mathbb{R}^p$ are outputs of a linear time invariant dynamical system driven by a one dimensional sparse and bounded input, $u(t) \in \mathbb{R}$. The dynamical system is defined by its system matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times 1}$, $C \in \mathbb{R}^{p \times n}$, a state vector, $x(t) \in \mathbb{R}^n$. Above, $n$ is the order of the LDS and $p$ is the dimension of the observation. This model can be expressed as follows:

$$\begin{align*}
\|u(\cdot)\|_{\ell_0} &\leq k \\
|u(t)| &\leq 1 \quad \forall t \\
\sum_i \|CA^iB\|_2^2 &\leq \mu \\
x(t+1) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t).
\end{align*}$$

(4.1)

where $\|u\|_{\ell_0}$ is the number of nonzero elements in the input sequence $u$. We can write (4.1) as:

$$\begin{align*}
\|U\|_{\ell_0} &\leq k \\
|u_i| &\leq 1 \quad \forall i \\
\sum_{i=0}^{N-2} \|CA^iB\|_2^2 &\leq \mu \\
Y &= \Gamma X_0 + HU.
\end{align*}$$

(4.2)
where $i$ is the discrete index, $N$ is the length of the signal, $Y = [y_0^T, y_1^T, \ldots, y_{N-1}^T]^T$, $U = [u_0, u_1, \ldots, u_{N-1}]^T$, $X_0 \in \mathbb{R}^n$ is the initial condition of the system, and

$$
\Gamma = \begin{bmatrix}
C \\
CA \\
CA^2 \\
\vdots \\
CA^{N-1}
\end{bmatrix},
H = \begin{bmatrix}
0 & 0 & \ldots & 0 \\
CB & 0 & \ldots & 0 \\
CAB & CB & \ldots & 0 \\
\vdots & \vdots & \ddots & 0 \\
CA^{N-2}B & CA^{N-3}B & \ldots & 0
\end{bmatrix}.
$$

Our representation consists of a total of 7 systems in the form of (4.2) that model body pose along with global position and orientation. Pose systems are learned from Euler angles grouped into 5 multidimensional time series for the body (torso, 2 arms, and 2 legs). The other 2 systems are learned from absolute 3D positions and orientation angles respectively.

The intuition behind the sparsity constraint on the input is to limit our solution space in such a way as to force as much of the stationary dynamics as possible into the system parameters. This way we hope to view our input as a triggering mechanism, a spike train sequence, that is a characteristic signature of the action. As shown in Fig. 4.1, the inputs found by our method given these constraints typically consist of sequences of impulses. It can thus be said that our representation interprets actions as a superposition of impulse responses.

One more aspect of our model is bounding the input. This forces variables such as the amplitude of an action into the system matrices, thus resulting in inputs that are more comparable across actions and individuals. Moreover, the unit impulse response of the system is bounded to prevent degenerate solutions due to scale ambiguity: $\|H\|_2 \rightarrow \infty$ and $\|u\|_1 \rightarrow 0$. 

40
4.3 Identification with Sparse Bounded Input

Instead of seeking to minimize the one-step prediction error, as in HMMs and autoregressive models, we focus on the full simulation error:

\[
\begin{aligned}
\text{minimize} & \quad \|\hat{Y} - \Gamma X_0 - HU\|_2^2 \\
\text{subject to:} & \quad \|U\|_{\ell_0} \leq k \\
& \quad |u_i| \leq 1 \quad i = 0, \ldots, N - 1 \\
& \quad \sum_{i=0}^{N-2} \|CA^i B\|_2^2 \leq \mu 
\end{aligned}
\]

(4.3)

where \(\hat{Y}\) is the observed time series. It is well known that the minimization in (4.3) is NP-hard, thus we relax the problem to a weighted \(\ell_1\) minimization [Tib96]:

\[
\begin{aligned}
\text{minimize} & \quad \|\hat{Y} - \Gamma X_0 - HU\|_2^2 + \lambda \sum_{i=0}^{N-1} w_i |u_i| \\
\text{subject to:} & \quad |u_i| \leq 1 \quad i = 0, \ldots, N - 1 \\
& \quad \sum_{i=0}^{N-2} \|CA^i B\|_2^2 \leq \mu. 
\end{aligned}
\]

(4.4)

The form above adds a regularizer term, with \(\lambda\) serving as the tradeoff between accuracy of fit and sparsity.

4.3.1 Alternating Minimization

Our approach to solve (4.4) is similar in spirit to algorithms for learning dictionaries to sparsely represent images [OF96]. In our case, however, the dictionary is the impulse response of the linear dynamical system \(H\).
Algorithm:

1. Select the order of the system\(^1\): \( n \)

2. Initialize a random sparse input \( U \) satisfying the constraint: \( |u_i| \leq 1, \ \forall i \)

3. Repeat

   (a) Given \( U \):

   Identify a LDS: \( A, B, C, X_o \)

   Scale \( B = \min \left( 1, \frac{\mu}{\sum_{i=0}^{N-2} \|CA^iB\|_2^2} \right) \cdot B \)

   (b) Given \( X_o, \Gamma \) and \( H \):

   \[
   \minimize_U \| \hat{Y} - \Gamma X_0 - H U \|_2^2 + \lambda \sum_{i=0}^{N-1} w_i |u_i| \\
   \text{subject to:} \quad |u_i| \leq 1 \quad i = 0, \ldots, N - 1
   \] (4.5)

For estimating the \( A \) and \( C \) matrices of the LDS we use the subspace identification algorithm for deterministic systems [VD96] with the constraint that \( A \) must be stable. For this purpose we adopt the method [SBG07], which incrementally adds constraints to a quadratic program to improve the stability of the estimated system matrix. Having estimated \( A \) and \( C \), the estimation of \( B \) and \( X_0 \) is the least-squares solution of the simulation error [CP01].

### 4.3.2 Enhancing Sparsity

The sparsity of the result obtained by solving a uniform weighted \( \ell_1 \) - regularized least-squares formulation (4.5) can be further enhanced by incorporating an iterative

\(^1\)the order of the system is verified during system identification [VD96].
reweighting scheme [CWB07]. Step 3(b) of the algorithm above is thus modified as follows:

1. Initialize the weights: \( w_i^{(0)} = 1, \ i = 0, \ldots, N - 1 \).

2. Solve the weighted \( \ell_1 \) minimization problem

\[
U^{(l)} = \arg\min \| \hat{Y} - \Gamma X_0 - H U \|_2^2 + \lambda \sum_{i=0}^{N-1} w_i |u_i| \\
\text{subject to: } |u_i| \leq 1 \ i = 0, \ldots, N - 1
\]

3. Update the weights: \( w_i^{(l+1)} = \frac{1}{|u_i^{(l)}|+\epsilon} \).

4. Terminate on convergence or when \( l = l_{\text{maxiter}} \).

4.4 Large Scale \( \ell_1 \) minimization

Estimating the input of a linear time-invariant (LTI) system, \( U \), using \( \ell_1 \) regularization is computationally intensive and becomes a challenging problem when the length of our observation is several thousand samples. However, we can reduce the computational cost significantly by exploiting the Toeplitz structure of the problem. The multiplication of a Toeplitz matrix with a vector can be performed in \( O(N \log N) \) instead of \( O(N^2) \). In our experiments we use the truncated Newton interior-point method proposed by Kim et al. [KKL07], modified according to the specific constraints of our formulation. In the situation where the output is multivariate, \( H \) can be represented with \( p \) Toeplitz matrices to maintain efficiency during multiplication.
Figure 4.1: This figure summarizes how our model captures actions and compares the representational power of sparse input driven LDS with that of traditional stochastic LDS for non-stationary actions. The top plot illustrates the input to the inferred LDS that drives a person’s right leg during a dance. The output of the right leg LDS corresponds to the 9 dimensions of the original joint angle time series. In the center plot we show that over the course of the dance we capture the joint angle of the right hip, one of the leg’s dimensions, with a median error of 3.57 degrees and a mean absolute error of 4.61 degrees with a standard deviation of 3.98 degrees. The original signal is shown in blue and our corresponding synthesis is shown in red. In the bottom plot the synthesis result of an LDS driven by Gaussian noise is shown with a solid black line. The dashed red line shows the 5-step prediction of the same stochastic system, and the original signal appears in blue.
4.4.1 Primal and Dual Problem

In order to use the duality gap to establish convergence criteria for the minimization, we derive the dual problem here. Our initial problem is:

\[
\begin{align*}
\text{minimize} & \quad \|HU - \tilde{y}\|_2^2 + \lambda \sum_{i=0}^{N-1} w_i |u_i| \\
\text{subject to:} & \quad |U| \preceq 1
\end{align*}
\] (4.6)

where \(\tilde{y} = \hat{Y} - \Gamma X_0\). We change variables \(\tilde{u}_i = w_i u_i\) and introduce the diagonal matrix \(D = \text{diag}(w)\), to transform to an unweighted \(\ell_1\) regularized problem, where \(w = [w_0, \ldots, w_{N-1}]^T\). Afterward, we introduce a new variable \(z \in \mathbb{R}^N\), a new equality constraint \(z = HD^{-1} \tilde{u} - \tilde{y}\), and make the box constraints implicit [BV04].

\[
\begin{align*}
\text{minimize} & \quad z^T z + \lambda \sum_{i=0}^{N-1} |\tilde{u}_i| \\
\text{subject to:} & \quad z = HD^{-1} \tilde{u} - \tilde{y}
\end{align*}
\] (4.7)

The dual function of (4.7) is:

\[
g(\nu) = \inf_{-w \leq \tilde{u} \leq w, z} (z^T z + \lambda \|\tilde{u}\|_1 + \nu^T (HD^{-1} \tilde{u} - \tilde{y} - z))
\]

\[
= \frac{\nu^T \nu}{4} - w^T ((D^{-1} H^T \nu + \lambda 1)^- + (D^{-1} H^T \nu - \lambda 1)^+)
\]

where \(q_i^+ = \max(q_i, 0)\), \(q_i^- = \max(-q_i, 0)\). Any dual feasible point \(\nu\) gives a lower bound on the optimal value of the primal problem (4.7).
4.4.2 Truncated Newton Interior-Point Method

The $\ell_1$ regularized least-squares problem (4.7) can be transformed to a convex quadratic problem, with linear inequality constraints.

\[
\text{minimize}_{\tilde{u},v} \quad z^T z + \lambda \sum_{i=0}^{N-1} v_i \\
\text{subject to:} \\
z = HD^{-1}\tilde{u} - \bar{y}; \quad -w \preceq \tilde{u} \preceq w; \quad -v \preceq \tilde{u} \preceq v.
\]

(4.8)

In this part we incorporate an interior-point method for solving our convex optimization. We first define the logarithmic barrier for the bound constraints in (4.8):

\[
\Phi(\tilde{u}, v) = -\sum_{i=0}^{N-1} \log(v_i + \tilde{u}_i) - \sum_{i=0}^{N-1} \log(v_i - \tilde{u}_i) \\
- \sum_{i=0}^{N-1} \log(w_i + \tilde{u}_i) - \sum_{i=0}^{N-1} \log(w_i - \tilde{u}_i).
\]

(4.9)

The central path consists of the unique minimizer $(x^*(\tau), v^*(\tau))$ of the convex function as the parameter $\tau$ varies from 0 to $\infty$:

\[
\phi_\tau(\tilde{u}, v) = \tau\|HD^{-1}\tilde{u} - \bar{y}\| + \tau\lambda \sum_{i=0}^{N-1} v_i + \Phi(\tilde{u}, v).
\]

(4.10)

In order to minimize $\phi_\tau(\tilde{u}, v)$, the search direction is computed as an approximate solution to the Newton system, using Preconditioned Conjugate Gradient [KKL07].

4.5 Experimental Evaluation

4.5.1 Datasets

The first dataset used is the FutureLight action dataset [SAM07], presented at Section 3.5.1. We applied our learning algorithm to the full joint angle representations of
all 158 samples in the dataset. In all cases we used models of order $n = 10$, with the exception of “Sit” actions which were estimated with order $n = 8$ due to the small number of available frames. We performed the deconvolution using the sparsity enhancing reweighting scheme with $\lambda = 10$ and $\epsilon = 0.005$. We use FutureLight in Section 4.5.2 and Section 4.6.3 to demonstrate accuracy and explore the supervised classification task.

To test our hypothesis about capturing action signatures in the inferred input we also obtained 6 long sequences from the CMU Motion Capture Database\textsuperscript{2}, in each of which a single actor performs several actions in succession. For instance, subject 86, sequence 3, contains smooth transitions between a number of sports related actions including walking, running, jumping, kicking, stretching, and even jump-kicking. We used the same parameter settings as in the FutureLight dataset ($n = 10$). In Section 4.6.2 we show that by taking simple statistics on the inferred input we were able to accurately classify the actions performed and localize their transitions (Fig. 4.3).

\subsection{4.5.2 Accuracy and Compression}

The least requirement for a model is that it captures the statistics of the data with smaller complexity than the data itself. We show that our model achieves this task by assessing the accuracy of our reconstruction and sparsity of the inferred input.

First, we show some qualitative results of a complex non-stationary dance sequence in Fig. 4.1. From Fig. 4.1, it is clear that our model captures motions accurately where typical Gaussian noise driven LDSs of similar complexity experience a significant lack of representational power. For a more extensive evaluation, in Table 4.2, we report the error in representing the position $(X, Y, Z)$ and joint angles, expressed in Euler angles, for the 5 actions in the FutureLight dataset. Further, in Table 4.1, we compare the

\textsuperscript{2}We used $\sim$5000 frames from subject 86, sequences 1, 2, 3, 5, 6, 7.
mean reconstruction error for these joint angles modeled with different approaches. Our model (\( Y \) (eq. 4.2)) achieves the smallest reconstruction error, illustrating that it captures the signal more accurately than Gaussian noise driven LDS, even when the latter systems are given the added benefit of using information from 5 time steps in the past.

Table 4.1: Comparison with other methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean Absolute Error (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model (Simulation)</td>
<td>4.96</td>
</tr>
<tr>
<td>Stoch. LDS (Simulation)</td>
<td>17.70</td>
</tr>
<tr>
<td>Stoch. LDS (5-step prediction)</td>
<td>5.29</td>
</tr>
<tr>
<td>Stoch. LDS (10-step prediction)</td>
<td>6.74</td>
</tr>
</tbody>
</table>

In addition to modeling individual actions well, we observed that in the CMU sequences our model could capture successive actions and their transitions with a single dynamic system. Typically, such transitions between actions are difficult to capture and historically have even been treated as independent action classes. Results using our inference on these sequences are discussed in Section 4.6.2 and Fig. 4.3.

Even though synthesis is a valid method of evaluating what we capture, it is not the key goal of our model. Thus we do not focus on adding any kinematic or smoothness constraints, as is often done in graphics literature [WP95, AFO03] to generate lifelike motions.

Finally, we compute that on average, in FutureLight, 78.84% of the input signal values are zero, confirming that the inferred signal is sparse. An advantage that comes with using our sparse input LDS representation is the compressive quality of the models. For a leg, whose original \( N \)-length time series has 9 dimensions, this representa-
Table 4.2: Representational power of our model as evaluated on the FutureLight dataset. The errors for angle measurements are in degrees. The 3D position errors are reported in the units of the motion capture data (inches scaled by a factor of 0.45). 

<table>
<thead>
<tr>
<th></th>
<th>Dance XYZ</th>
<th>Dance XYZ</th>
<th>Jump XYZ</th>
<th>Jump XYZ</th>
<th>Sit XYZ</th>
<th>Sit XYZ</th>
<th>Run XYZ</th>
<th>Run XYZ</th>
<th>Walk XYZ</th>
<th>Walk XYZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean Absolute Error</td>
<td>1.38</td>
<td>4.50</td>
<td>1.50</td>
<td>5.83</td>
<td>1.63</td>
<td>5.89</td>
<td>1.46</td>
<td>5.71</td>
<td>1.55</td>
<td>1.90</td>
</tr>
<tr>
<td>Standard Deviation of Absolute Error</td>
<td>1.05</td>
<td>3.69</td>
<td>1.13</td>
<td>5.01</td>
<td>1.21</td>
<td>4.84</td>
<td>1.31</td>
<td>4.75</td>
<td>1.74</td>
<td>2.38</td>
</tr>
<tr>
<td>Median of Absolute Error</td>
<td>1.16</td>
<td>3.66</td>
<td>1.28</td>
<td>4.64</td>
<td>1.37</td>
<td>4.72</td>
<td>1.15</td>
<td>4.61</td>
<td>1.08</td>
<td>2.38</td>
</tr>
</tbody>
</table>


tion typically reduces to only 8.4% of the original size across all actions (not counting the constant overhead to store the dynamical system parameters).

4.6 Applications

Aside from providing a concise and accurate representation of complex actions, our model allows for a number of other applications. In this section, we outline possibilities of how our model can be leveraged, and explore our hypotheses regarding the information encoded by the inferred input.

4.6.1 Creative Synthesis of Actions

Once a model is learned, the inputs and parameters of each dynamical system are directly available and can be controlled purposefully in ways similar to Doretto et al. for dynamic textures [DS03]. For example, we can change the intensity of the motion by scaling the $C$ matrix of the system. This type of creative editing of the dynamics and input can result in interesting variations on an original action. Examples are illustrated at [http://vision.ucla.edu/~mraptis/spikes](http://vision.ucla.edu/~mraptis/spikes).
Figure 4.2: A spinning ballet dance sequence (sample 19 from FutureLight). The original sequence is shown in the top figure, and the synthetic sequence generated from our sparse input LDS is shown in the bottom figure. Notice that the curved trajectory of the dance as well as the complicated articulation of the body were both captured closely by our model.

4.6.2 Unsupervised Action Segmentation

Having observed that our model is strong enough to capture transitions between distinct actions, we can pose the inverse problem of detecting such transitions from data. For this we use the six long sequences from the CMU dataset. Since we constrain the dynamics to be constant throughout a single sequence, finding transitions corresponds to temporal segmentation of the input. We go one step beyond simple segmentation and identify repeated patterns in the input that correspond to distinct actions. With this
experiment we therefore test the hypothesis that the input signal captures signatures of observed actions.

To segment and classify the spike-trains we construct histograms of input signal intensities in a window around each frame, capturing the local statistics. Each histogram is quantized into 11 bins, equally spaced in a range from -1 to 1. To encode information from the inputs to all of the body’s systems the histograms for each limb and torso are stacked to create a frame descriptor. We then use the lossy coding approach of [MDH07] to produce an unsupervised segmentation. To encourage temporal coherence, we initially restrict merging to only neighboring segments, using a low $\rho$ distortion parameter ($\rho = 25$). This also significantly reduces the computational cost. After convergence of the first merging the neighbor restriction is removed in order to cluster repeating actions. For this final clustering we look for the two most stable segmentations across a range of $\rho$ values ($\rho \in [30, 130]$) and, of those, select the segmentation with the minimum $\rho$. Given the input identified by our algorithm, this full segmentation procedure takes approximately 1 minute to run on a 5000 frame sequence. Our results are shown in Fig. 4.3.

Fig. 4.3 also provides a comparison of our input-based segmentation results with existing algorithms for segmenting temporal data that operate directly on observed values. We compare with Barbič et al. [BSP04], who proposed a change detection algorithm based on the reprojection error on the principal components computed in a sequential fashion. Their algorithm detects changes in motions relatively accurately, however it does not cluster the distinct actions. The method of Vidal [Vid08] models the first two principal components of the data as a first order Switched Autoregressive Exogenous Model and identifies the model’s coefficient recursively. Change of the model’s coefficient implies change of human motion. Similarly, Ozay et al. [OSC08] model the first three principal components of the data as a third order switched autore-
gressive model with piecewise constant coefficients. The coefficients are then clustered with $K$-means (with $K$ manually selected to the optimal number for each sequence). The segmentation results of both models are shown in Fig. 4.3 for comparison.

As a quantitative measure of the performance of our unsupervised clustering, we compared the areas of the regions segmented with the areas of the ground truth segmentation provided by [BSP04]. Since transitions between actions are typically smooth, (thus there are no “true” transition instants), labels assigned to regions ground truthed as transitions are not counted towards or against the classification score. Labels assigned by our algorithm were considered correct when they matched across repeated actions within a sequence and when they were unique for actions appearing only once. With this metric within-class oversegmentation does not count against us as long as it is consistent when that action class is repeated (this is observed in the “Rotate Body” actions in sequence 7). Averaging over the 6 sequences, we obtained a mean classification rate of 90.94\% according to the above metric. For the SAR method the mean classification rate was 72.27\%. Our result illustrates that the distinct complex patterns of the observed data were accurately captured as patterns of the sparse input signals.

4.6.3 Supervised Classification

Without any supervision we deconvolve the 158 samples of the FutureLight dataset. We then classify our observations using only statistics of the input signals defining the relative pose of the actor (the actor’s global position and orientation are neglected). We extract features for each sparse signal with a sliding window of length 50 and step size of 16. Within each window the features we extract include the percentage of zero elements, the percentage of successive non-zero elements that maintain the same sign, as well as the percentage of successive non-zero elements that change sign.

A dictionary is created from the extracted features with $K$-means clustering ($K =$
Table 4.3: Confusion Matrix of Future Light Dataset. Overall mean performance 83.87%.

<table>
<thead>
<tr>
<th></th>
<th>Dance</th>
<th>Jump</th>
<th>Sit</th>
<th>Run</th>
<th>Walk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dance</td>
<td>24</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Jump</td>
<td>2</td>
<td>11</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sit</td>
<td>1</td>
<td></td>
<td>34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>3</td>
<td>3</td>
<td></td>
<td>23</td>
<td>1</td>
</tr>
<tr>
<td>Walk</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>43</td>
</tr>
</tbody>
</table>

12), and each extracted window is projected onto the dictionary. At this stage, each of the 5 input signals representing an actor’s body is translated into a sequence of labels. To take into account the temporal alignment of labels but also utilize support vector machines (SVM) with RBF kernel, we use the Smith-Waterman based technique described in Section 3.4.3. Classification performance is evaluated with a leave-one out cross validation approach, as has been used throughout the literature. We illustrate our classification results in Table 4.3, and compare with other methods in Table 4.4.

These results confirm that basic features of the sparse input signals capture characteristics of the observed time series. In this scenario, patterns were found to be characteristic of action classes despite inference of model parameters taking place independently for each action example. We achieve reasonably good performance on this task, but do not quite match discriminative approaches that construct dictionaries directly on the multi-dimensional observation signals. However, we make the point that our model generalizes to other tasks such as segmentation and synthesis, all of which it performs in a satisfactory fashion, while the other methods lack these capabilities. We also suspect that performance on supervised classification could be further
Table 4.4: Comparison of classification results.

<table>
<thead>
<tr>
<th></th>
<th>FutureLight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 3 [RWS08]</td>
<td>98.03</td>
</tr>
<tr>
<td>[SAM07]</td>
<td>89.7</td>
</tr>
<tr>
<td>Spike Train Classification</td>
<td>83.63 ± 1.23</td>
</tr>
</tbody>
</table>

boosted by incorporating prior knowledge of the action classes into the deconvolution procedure, however we leave this investigation for future work.

4.7 Conclusion

In this chapter, we have proposed a new and efficient alternating minimization algorithm for blind identification of linear dynamical systems driven by sparse inputs. By applying our model to a wide range of publicly available motion capture data, we have shown that this new class of models is powerful enough to capture non-stationarities of human motions. Finally, through both supervised and unsupervised segmentation and classification experiments we have demonstrated that our model is able to capture characteristic signatures of the observation in the inferred inputs. This makes it useful for analyzing sequences of various actions and applications where temporal ordering and representational accuracy are important.

Although we use motion-capture data to evaluate our dynamical models, the ultimate goal is to use these models to infer and classify time sequences of video, both at the low-level (detection and tracking) and at the high-level (action recognition).

Algorithms: STS: Spike Train Segmentation (our Method), PCA: [BSP04], HUMAN: ground truth segmentation, RSARX: [Vid08], SAR: [OSC08]

Figure 4.3: [Best viewed in color] Here we show the performance of our temporal segmentation/action recognition on complex CMU MoCap data (subject 86, sequences 1,2,3,5,6,7; shown left to right, top to bottom). Colors correspond to label values, thus regions marked with the same color are those that have been clustered as the same action. In sequence 2 we notice that transition regions are often identified as unique. This is due to the fact that transition regions are often smooth and do not exhibit regular statistics like their neighboring actions. In sequence 3 we notice that “walk” and “run” are confused. Likewise in sequence 5 there is confusion between “jump up”, “jump forward”, and “jump on one leg”. In sequence 6 we oversegment “run” into 2 different labels, however neither of these are confused with any other action in the sequence. Finally the most interesting and exciting result is sequence 7. Here we have oversegmented the “rotate body” actions, but have broken them down into very regular components. We see that each instance of “rotate body” is composed of a starting transition (brown), two alternating short actions (light blue, dark blue), and an ending transition (pink).
CHAPTER 5

Real-Time Classification of Dance Gestures from Skeleton Animation

5.1 Introduction

Real-time depth sensing systems have recently sparked a revolution in the videogame industry by embracing the human body as a controller. In this chapter, we will focus on developing an action recognition algorithm that inherits the advantages of long temporal range features, similar to the ones proposed in Chapter 3 and 4, and is designed to be robust to the noisy measurements that the depth sensing systems have. The XBOX Kinect system, an example of such a natural user interface, parses a depth-map stream at 30 frames per second to estimate in real-time the positions of 16 predefined points that constitute a wireframe skeleton of a moving user. Subsequent algorithmic processing can then attempt to understand the user’s motion in order to interactively control gameplay.

Dancing is a quintessential form of human motion and expression; hence, it deserves special focus within such a computing platform [KPS03, HGP04]. Compared to existing experiences such as Harmonix’ Dance Central [Har] where a player, vis-à-vis, repeats the motion posed by an animated character in the videogame, our aim is to expand substantially the interaction that the dancer has with the avatar animation and control, by allowing him/her to dance at any time any of the pre-choreographed
gestures that are modeled in system’s database. To address this objective, our system learns a statistical model that captures the nuances of a predetermined set of gesture classes, and then uses the model to classify the input skeletal motion. The system is comprised of several modules as illustrated in Fig. 5.1.

A novel **angular skeleton representation** is a key to overall system performance. It is used to map the skeleton motion data to a smaller set of features (each a scalar time series) that can be robustly estimated from the noisy input and yet retains the salient aspects of the motion. The aim is to reduce the overall entropy of the signal, remove dependence on camera position, and avoid unstable parameter configurations such as near gimbal lock. The approach is to fit the full torso with a single frame of reference, and to use this frame to parameterize the orientation estimates of both the first- and second-degree limb joints.

A **cascaded correlation-based max-likelihood multivariate classifier** takes into account the fact that dancing adheres to a canonical time-base, a musical beat, to simplify the template matching process. During training, our classifier builds a statistical model for each gesture class based upon both an oracle and a database of gesture instances performed by a group of subjects with a wide range of dancing skills. At runtime, the classifier correlates the multivariate input buffer with the prototype gesture model for each class and constructs a per-class log-likelihood score. Then, it uses the scores to rank all classes and performs rounds of logistic regression tests among the top classes to identify the winning match.

A **space-time contract-expand distance metric** is the final step in our approach; it uses dynamic time-warping with exponential scaling of time-space to achieve robust comparison of the input gesture with the matched oracle. The difficulty in this task stems from the noise present in the skeletal data and the fact that humans exhibit a wide spectrum of ability to replicate a specific motion.
We test our system on a large corpus of data acquired using the XBOX Kinect platform. It comprises 28 choreographed gesture classes, and hundreds of gesture instances performed by dozens of people for each gesture class. Our classifier attains a classification accuracy of 96.9% on average for approximately 4-second skeletal recordings. The computational complexity of the detector is linearly proportional to the number of classes in the system. For a 4-second input buffer, it consumes only 2 and 25 milliseconds on a state-of-the-art PC and a single XBOX core respectively (and the present implementation does not yet use a vectorized FFT implementation). Classification accuracy appears to scale well to a larger database. We also observe strong robustness to various types of noise including mislabeling in the training data.

5.2 Related Work

Controlling and directing a virtual character is an important problem in computer graphics movies and computer games. Several different approaches have been pro-
posed for the creation of realistic avatar motions that satisfy user constraints [BH00]. 
[AF02, KGP02] creates a graph structure to effectively search for human motions that 
satisfy low-level user constraints, e.g. a particular key-pose in a particular time in-
stance. [LCR02] organizes a motion capture database into a graph for efficient search, 
and generates motion using three different interfaces: a list of choices, a sketch-based 
interface and a video performance. The vision-based control matches silhouette fea-
tures of the user with the features obtained by rendering the motion capture database 
at each video frame. The recent system of [IWZ09] takes an input feed from a motion 
capture system and translates the human intention into an action in the virtual world. 
Their proposed action recognition method is based on the reconstruction error of body 
poses in a low-dimensional space. This limits the classification ability to motions that 
differ significantly, such as walking versus climbing. Real-time depth-sensing sys-
tems enable us to do human motion analysis in a rather unconstrained environment. 
In turn, this lets us go beyond simple avatar control, and enables the use of semantic 
information in the interface.

In this section, since the current chapter is mainly focused on 3-D joint trajec-
tories extracted from the depth image sequence, we will explore in more depth action 
recognition algorithms that have as input 3-D point trajectories. Campbell and Bo-
bick [CB95] represent the 3D joint trajectories as a collection of 2D curves in phase 
spaces, and performs action recognition based on the distance to each of these curves 
at each time instance. This does not take into account the dynamics of the activity. 
In the work of Lv et al. [LNL05], actions are represented by spatiotemporal motion 
templates and multi-scale template matching is performed to deal with possible tem-
poral scale changes. However, the proposed motion templates do not account for phase 
alignment of the human actions. Such alignment is a crucial component of our algo-
algorithm. Moreover, the classification algorithm presented in [LNL05] does not consider 
the joint distribution of the motion features. The latter is important for identifying
complex dancing moves. The demand for motion retrieval from large motion capture datasets for the purpose of motion synthesis has led to the adoption of efficient time series techniques. Faloutsos et al. [FRM94] introduce a subsequence-matching method that maps motion sequences into a small set of multidimensional rectangles in feature spaces. Keogh et al. [KPZ04] present a low-complexity technique for similarity search that takes into account possible intrinsic temporal variations, based on a lower bound [Keo02] of dynamic time warping. Forbes and Fiume [FF05] develop a weighted principal component analysis (PCA) based representation for human poses, mapping each human motion to a low-dimensional parametric curve, enabling efficient retrieval. Similarly, Liu et al. [LZW05] introduce a indexing method that identifies a subset of discriminative feature using PCA. The distance function of [OFH08] shows the discriminative power of features based only on the kinetic energy of the motion signals. Accordingly, our recognition algorithm decomposes signals into correlation and energy features. Müller et al. [MRC05, MR06] transform motions into time-varying geometric feature relationships to achieve low dimensionality and robustness in the matching process. Their approach is scalable and efficient for noise-free motion capture data. Nevertheless, it is a nontrivial problem to define geometric (semantic) relationships that are discriminative and robust for highly complex human motions, e.g. dancing.

5.3 Skeletal Representation

The current version of the skeletal tracking algorithm (STA) [SFC11] in the Kinect platform identifies a wireframe skeleton at a rate of 30fps for each player silhouette recognized in the sensed datastream. As illustrated in Fig. 5.2, the skeleton is represented by 16 points with the root of the cartesian 3D coordinate system positioned in the neck and oriented in alignment with the sensor. Each point is marked as “tracked”
or “inferred” – the latter typically denoting the estimate of an occluded body part. Inferred joints are commonly plagued with substantial noise.

Our goal is to infer from this skeleton a set of features that enable effective recognition of dance gestures. These features do not require the completeness or precision necessary for re-rendering [KOF05]. One can understand the intuition behind the importance of feature construction in machine learning by noticing that the original signal may, and typically is likely to experience loss of entropy when dimensionality-reduced to a feature vector. Once information has been lost, no classifier can recover it. Conversely, if the features are presented in a format that follows some distribution well-addressed by a group of classifiers, all will perform relatively equally well. Thus, we seek a specialized set of objectives:
• **Redundancy reduction** – express wireframe joints relative to their parent nodes, like in traditional joint-angle representations, as heuristically this should eliminate the redundancy of the resulting multivariate time-series (see Fig. 5.3).

• **Robustness** – overcome data errors, which are more challenging in real-time depth sensors compared to modern motion capture systems due to two factors: first, there exists strong additive noise intrinsic to the sensing system that propagates through the STA into the resulting skeleton and second, occluded parts of the skeleton are inferred and thus prone to substantial errors.

• **Invariance to sensor orientation** – maximize the invariance of the skeletal representation with respect to camera position.

• **Signal continuity and stability** – orient the coordinate axes used to compute relative positions so as to minimize the probability of signal discontinuities, e.g., gimbal lock. This objective is particularly important as we proceed to use normalized correlation for gesture detection.

• **Dimensionality reduction** – reduce the dimensionality of the search space for classification while retaining the character of the motion; compared to representations that focus on animation or motion capture, our interest lies predominantly in computing features that may not be perfectly invertible, but offer simplicity, intuitive resulting time-series, and performance.

5.3.1 **Torso PCA frame**

We observe that the points of the human torso (defined by 7 skeletal nodes as illustrated in Fig. 5.2) rarely exhibit strong independent motion. Thus, the torso can be treated as a vertically elongated rigid body. Yet, due to the strong noise patterns in the depth
sensing system, we observe that individual torso points, in particular shoulders and hips, may exhibit unrealistic motion that we would like to limit rather than propagate by relative representation. Consequently, we aim to treat the torso as a rigid body with all of its points contributing to the estimate of its position, then use this estimate to represent the remainder of the human skeleton in relative manner.

We compute the principal components for the torso points, i.e., a 3D orthonormal basis as a result of applying PCA to the 7-by-3 torso matrix. The first principal component $u$ is always aligned with the longer dimension of the torso, and we can canonically orient it (top-down) because in most dancing it is not anticipated that the player’s torso will stand upside-down relative to the sensor. In contrast, for the second principal component $r$, aligned with the line that connects the shoulders, the orientation is not so easily inferred, and here we must rely on the “left-right” skeleton orientation inferred by the STA. Finally, the last axis of the orthonormal basis is computed as a cross prod-
Figure 5.4: Illustration of torso PCA frame, and its use in defining spherical coordinate systems for first- and second-degree joints.

The product of the first two principal components, i.e., \( t = u \times r \). We call the resulting basis \( \{u, r, t\} \) the torso frame.

The torso frame is well aligned with our previously stated objectives. It is an exceptionally robust and reliable foundation for a coordinate system based upon the orientation of the human body. Although it is dependent upon camera position, points represented within a coordinate system that is derived from the torso frame may be fully invariant to the sensor. It reduces 7 3-D trajectories of the original problem specification to a new set of signals whose aim is to describe only the 3D orientation of the resulting orthonormal basis. We detail in Section 5.3.4 a set of simple features that intuitively and robustly describe torso’s motion. Finally, we add a brief note that it might be possible to compute the torso frame more accurately from the underlying depth-map silhouette. Our speculation is that the computational overhead of such an approach does not offer a favorable trade-off with respect to the ensuing minor improvement in recognition performance.
5.3.2 First-Degree Joints

We denote all joints adjacent to the torso as first-degree joints – these include elbows, knees, and the head. We represent these points relative to the adjacent joint in the torso in a coordinate system derived from the torso frame. Consider the illustration in Fig. 5.4(center), where $LE$ – the left elbow, is represented relative to $LS$ – the left shoulder. First, we translate the torso frame, $\{u, r, t\}$, to $LS$ and construct a spherical coordinate system such that the origin is centered at $LS$, the zenith axis is $u$, and the azimuth axis is $r$. Then, $LE$’s position is described by

- its **radius** $R$ – the distance of $LE$ from the origin,
- its **inclination** $\theta$ – the angle between $u$ and $\overrightarrow{LS, LE}$, and
- its **azimuth** $\varphi$ – the angle between $r$ and $\overrightarrow{LS, LE_p}$ where $LE_p$ is the projection of $LE$ onto the plane whose normal is $u$.

Since the length of the humerus bone is normalized and constant, we ignore $R$ for any further consideration. Thus, using this representation model, each first-degree joint is represented with two angles $\{\theta, \varphi\}$.

5.3.3 Second-Degree Joints

We denote as second-degree joints the tips of the wireframe extremities; these include the hands and the feet. The most descriptive vector associated with a second-degree joint is the bone that connects the adjacent first-degree joint and its adjacent torso joint. For example, as illustrated in Fig. 5.4(right), a vector $b$ protruding out of the humerus bone is a great candidate for the zenith direction of a spherical coordinate system with an origin in the left elbow, $LE$. Let’s denote the joint of the left hand as $LH$. Then, $LH$’s position is described by:
• its radius $R$ – the distance of $LH$ distance from the origin,

• its inclination $\theta$ – the angle between $b$ and $(LE, LH)$, and

• its azimuth $\varphi$ – the angle between $r_P$, the projection of $r$ onto the plane $S$ whose normal is $b$, and $(LE, LH_P)$ where $LH_P$ is the projection of $LH$ onto $S$.

Since the length of the forearm bone is normalized and constant, we ignore $R$. Thus, our model represents each second-degree joint using two angles $\{\theta, \varphi\}$. The consequences are equivalent to those of first-degree joints with one notable difference. While the inclination $\theta$ for second-degree joints is an exceptionally robust descriptor, their azimuth is not. Because the origin of the spherical coordinate system is not part of the rigid body that defines the torso frame, the orientation of $r$ is dependent upon the torso’s orientation and introduces noise into $\varphi$. In our experiments we have confirmed that this effect is rather mild and does not pose a challenge for the remaining steps of the classifier.

We identify an interesting problem: $b$ and $r$ could be oriented in such way that $b \cdot r = 1$, thus making the projection $r_P$ a point. While this is unlikely to occur,
any small angle between \( \mathbf{b} \) and \( \mathbf{r} \) is likely to pose increased levels of noise due to the instability of \( r_p \). Although this problem could be resolved in several ways, we observed in thorough experimentation that the case \( \mathbf{b} \cdot \mathbf{r} \approx 1 \) seldom occurs when \( \mathbf{r} \) is chosen as an azimuth reference. Note that instead of \( \mathbf{r} \) we could have used \( \mathbf{u} \) or \( \mathbf{t} \) or any linear combination of these vectors with a wide range of impact on final performance. We selected \( \mathbf{r} \) for one simple reason – its selection attenuated the issue sufficiently.

### 5.3.4 Feature Set

The novel angular wireframe model is represented by eight pairs of angles \( \{\theta, \varphi\} \) for each set of the four first-degree\(^1\) and four second-degree joints, as well as the rotation matrix of the torso frame with respect to the camera’s coordinate frame.

To parameterize the rotation matrix, we use Tait-Bryan angles (i.e., yaw, pitch and roll), which is simple and intuitive. Although Euler angles are prone to gimbal lock, this problem can be avoided using quaternions, but this approach yields rather unintuitive time-series data. In practice, because the STA does not support tracking of a player who is spinning, Tait-Bryan angles can be oriented so as to rarely introduce the feared gimbal lock. This is rather important for the normalized correlation scheme in our classifier.

We denote the set of feature time-series obtained from skeletal motion as \( \mathbf{f} = \{f_i(t), i = 1 \ldots 19\} \) and emphasize the fact that we reduced the complexity of our input from a collection of 16 3-D curves to a set of 19 1D vectors, a substantial simplification of the input specification with infrequent, negligible loss of information. Consequently, these features are geared for classification because they represent motion in relative manner that facilitates aligned, one-dimensional comparison.

\(^1\)In our current implementation, we ignored the head point and thus relied on just four first-degree points.
Fig. 5.5 illustrates a feature set obtained by recording a subject dancing the Clap-SlideNStomp gesture. One can observe that the torso is not too active during this gesture, the arms make two large up-down swings, each swing is preceded by a stomp, and that the dancer is moving sideways during the gesture.

In summary, our features convert 3D point trajectories to a set of time series that are robust to various types of noise common for skeletal tracking, experience exceptionally low loss of information, and are well positioned to generate “proper” distributions as input to classification tools.

5.4 Cascaded Classifier

The computed features of an acquired gesture are used as input to a classifier whose objective is to accurately identify which gesture class is best matched against the input. The flow of information during classification is illustrated in Fig. 5.1.

5.4.1 System Assumptions and Input/Output

A key assumption about the input to our classifier is that dancing adheres to a beat pattern – a fact rooted in the core skill and experience behind dancing. This allows us to ignore actual time and resample the input time-series so that within, say 8 beats, we create 120 frames of skeletal motion (a rate close to 30fps). This makes our classifier invariant to the pace of the beat in different musical pieces and eliminates the need to unwarp and synchronize different instances of players dancing the same gesture. Clearly, within our gaming scenario, it is assumed that each beat of music played during the game is labeled. Beat detection algorithms such as [TC02] could be used in this setting too.

We also assume that the player is allowed to dance only a limited, well-defined,
and known set \( M \) of \( K \) moves that span over 8 beats. This way we avoid on-line learning scenarios that could be detrimental to overall error rates. Incoming frames with skeletal data are stored in a FIFO buffer. Prior to classification, the contents are resampled at a rate of 120 frames per 8 beats. The classifier finds the best matched class in \( M \) and finally, responds with a report that outlines how well the player danced the matched gesture.

Figure 5.6: Feature vectors for the derived prototype mean and the oracle that correspond to the “ClapSlideNStomp” gesture.

5.4.2 Gesture Model

In training, we build a model of each choreographed gesture by relying on:

- a training set, \( F_T = \{ f_j, j = 1 \ldots L \} \) – a collection of \( L \) recordings of subjects
dancing this gesture; subjects of various skill participated in the recordings, each one typically producing a handful of recordings per gesture.

• an oracle, \( f_o \) – a recording of a gesture performed by a professional dancer. This recording is considered the definition of the gesture; a single or handful of recordings is considered mainly because professional dancers usually repeat a specific gesture so accurately that most of the variation in the recordings stems from sensor noise.

In order to produce the expected average trajectory of a dancer for each individual feature, denoted as a **prototype mean**, we first align the training data with respect to the oracle by computing a circular normalized cross-correlation between \( f_o \) and each individual \( f_j \). Circular normalized cross-correlation \( c \) of two vectors \( u \) and \( v \) is computed as:

\[
c(u, v) \equiv u \ast v \equiv \frac{(u(-t) - \overline{u}) \ast (v(t) - \overline{v})}{||u - \overline{u}||_2 ||v - \overline{v}||_2},
\]

where \( \overline{u} \) denotes the mean of \( u \). Circular cross-correlation of two vectors \( u \) and \( v \) can be computed as \( \mathcal{F}^{-1} [\mathcal{F}(u) \cdot \mathcal{F}(R(v))] \), where \( R() \) denotes reflecting the time-series vector and \( \mathcal{F} \) is the discrete Fourier transform. Normalized cross-correlation is computed for each feature. In order to account for the synch of the entire body, we sum the cross-correlation vectors for all features into a single vector \( \hat{c}_{j,o} = \sum_i c_{j,o}^i \).

The phase offset of the two vectors equals:

\[
\tau_j = \arg \max_t \hat{c}_{j,o}(t),
\]

thus we phase-shift all features in \( f_j \) for \( -\tau_j \) samples in order to align the \( f_j \) recording with \( f_o \).

We define a **prototype mean** for a specific feature as \( f_{m,i} = \frac{1}{L} \sum_{j=1}^{L} f_{j,i}(-\tau_j) \) and denote the gesture prototype as \( f_m \). The relation of \( f_m \) and \( f_o \) is that \( f_m \) represents
the motion of an *average subject* dancing the gesture, while $f_o$ is that of the expert. Typically they are similar in shape but the prototype mean is often attenuated in amplitude because skilled dancers usually emphasize movement for overall appeal. As an example, Fig. 5.6 illustrates $f_m$ and $f_o$ per feature for the “ClapSlideNSStomp” gesture.

Next, we assemble a model that captures the in- and out-of-class correlation statistics. For each recording $j$ in $F_T$ and feature $i$, we compute $c_{j,m}^i = f_{m,i} \star f_{j,i}$, $\tau_j' = \arg \max_t \sum_i c_{j,m}^i(t)$ and assemble for each feature $i$ a histogram of correlation values across $\{c_{j,m}^i(\tau_j'); j = 1 \ldots L\}$. Since $L$ is typically small, we apply a simple kernel density estimation (KDE) filter which smoothes the histogram using a gaussian kernel [Bis06]. We store the histogram curve for a specific feature $i$ as a lookup table, $p_i(c)$, where $-1 \leq c \leq 1$ is the correlation argument. For a particular feature, the lookup table thus returns the likelihood that given a correlation of the prototype mean and the input, the input gesture belongs to this specific class. Similarly, we can collect statistics on out-of-class correlations and create a corresponding lookup table $q_i(c)$. These two tables can now be combined to a scoring function for a specific correlation value. In our case, after much experimentation, we converged on: $h_i(c) = 2 \log(p_i(c)) - \log(q_i(c))$. Here, we must consider the fact that skilled dancers, i.e., dancers who produce high correlations against prototype means, are typically infrequent in $F_T$, which may result in low $p_i(c)$ for high $c$. In that case, their scores are essentially penalized for their dances being “too good”. To correct this anomaly, prior to applying the KDE filter, we adjust the histogram counts for high correlations.

As an illustration, in Fig. 5.7 we present the $p_i(c)$ and $q_i(c)$ curves for the “ClapSlideNSStomp” gesture. One can observe which features contribute more to the “uniqueness” of this gesture based upon their Kullback-Leibler –divergence.

Normalized cross-correlation as a detection technique [WTK87] is effective in matching shapes, but inefficient in matching their amplitude. Rather than using Eu-
Figure 5.7: Histograms $p_i$ and $q_i$ obtained over the training set of recordings for the "ClapSlideNStomp" gesture.

clodean distance or correlation without normalization, we opted for an additional distance metric, the average signal energy, as a complement to normalized correlation. Thus, for each feature $f_i$ of an in-class gesture instance, we compute its energy-level relative to the oracle’s as: $\alpha_i = ||f_{o,i}|| - ||f_i||$, then build a histogram $e^+_i(\alpha)$, $-4\pi^2 \leq \alpha \leq 4\pi^2$ over the energy-levels of all instances in $F_T$, and finally apply a KDE filter.

Similar to the correlation histogram $h_i(c)$, we compute the same statistic for out-of-class instances, $e^-_i(\alpha)$, combine them as $e_i(\alpha) = 2\log(e^+_i(\alpha)) - \log(e^-_i(\alpha))$, and finally compensate $e_i(\alpha)$ for the fact that skilled dancers, who are not common in our benchmark, may have wider range of motion and thus, increased energy level of their recordings. The latter adjustment is performed by increasing the histogram counts of
\( e_i^+(\alpha) \) for cases of low \( \alpha \). Thus, for a specific gesture and feature \( i \) our model encompasses a 3-tuple \( \{ f_{m,i}, h_i, e_i \} \) that consists of the prototype mean \( f_{m,i} \), the correlation histogram \( h_i(c) \), and the energy-level histogram \( e_i(\alpha) \).

### 5.4.3 Class Ranking

The previous subsection outlined the training procedure for our classifier; in this subsection we describe the real-time classification of user motion. The input to our classifier is a stream of skeletal wireframes that we convert to feature sets. Let \( x = \{ x_i, i = 1 \ldots 19 \} \) denote the input stream of 19 features, each \( N \) samples long. For each gesture model \( g = \{ \{ f_{m,i}, h_i, e_i \}, i = 1 \ldots 19 \} \) in our database we compute its score using the following steps.

- **normalized cross-correlation** – we first cross-correlate each input feature, \( x_i \), with its corresponding prototype mean, \( f_{m,i} \). This is the most computationally demanding operation of the classifier because radix-2 FFTs of length \( N \) are computed in \( \mathcal{O}(N \log(N)) \) operations.

- **max-likelihood score** – then we look up the corresponding histogram scores and sum them across all features:

\[
s = \frac{1}{19} \sum_{i=1}^{19} [h_i(c_i) + e_i(\alpha_i)].
\]  

(5.3)

- **phase-offset** – finally, we identify the phase shift \( \tau \) of the input relative to the prototype mean as:

\[
\tau = \arg \max_t s(t).
\]

(5.4)

The phase shifts are distinct for each class.

The **classification score** for each gesture class \( k \) in the database is \( s_k(\tau_k) \). We use these scores to rank all classes, with the best match having the highest score.
5.4.4 Pairwise Matching

The ranking classifier can still be improved because some classes are often similar in motion to the point where their prototype means across all features are equivalent except for one. One can view all instances of two gesture classes as a collection of points in a locality of a large $2(19 + 19)$-dimensional space. Due to acquisition noise and the variety of ways in which humans can play a certain gesture, two classes whose prototype means are nearly identical (across all but very few features) may have intersecting volumes if, for example, a multidimensional sphere is used to contain and detect all points of a specific class. Since the disambiguation of the two classes is more nuanced and selectively dependent upon features, there exists need to better distinguish neighboring classes using an advanced, pairwise matching tool.

Weighting of likelihoods in Equation 5.3 is one way to improve the classification agility. It is important to stress that the “optimal” weights need to be recomputed and are likely distinct for each pairwise comparison of gesture matches. Thus, we opt to compute these weights using standard logistic regression [Bis06] and deploy the trained coefficients at classification as follows:

**logistic regression** – for the two top-tiered classes with highest $s(\tau)$ scores, say indexed $k_1$ and $k_2$, we perform a binary classification by computing:

$$Pr(C = k_1 \mid \mathbf{x}) = 1/(1 + e^{-\gamma}),$$

$$\gamma = \sum_{i=1}^{19} w^{(k_1,k_2)}_{h,i} h_{k_1,i}(c_{k_1,i}) + w^{(k_1,k_2)}_{e,i} e_{k_1,i}(\alpha_{k_1,i}) +$$

$$\sum_{i=1}^{19} w^{(k_2,k_1)}_{h,i} h_{k_2,i}(c_{k_2,i}) + w^{(k_2,k_1)}_{e,i} e_{k_2,i}(\alpha_{k_2,i}),$$

(5.5)

where all weights have been trained using logistic regression. If $Pr(C = k_1 \mid \mathbf{x}) \geq 0.5$, class $k_1$ would be denoted as the best match, otherwise $k_2$. The process of pairwise matching the “winner class” with the next “runner-up class” could be repeated recur-
sively, although the likelihood that a class deep on the $s(\tau)$-list “wins” rapidly declines. Thus, in our experiments we use only a 3-deep sequence of pairwise class-comparisons via logistic regression.

Therefore, we augment our gesture model $\{f_{m,i}, h_i, e_i\}$ with another data field, the coefficient matrix for logistic regression:

$$W = \{w_{(k_r,k_q)}^{(k_r,k_q)}, w_{e,i}^{(k_r,k_q)}, w_{h,i}^{(k_q,k_r)} | i = 1 \ldots 19, r = 1 \ldots K, q = 1 \ldots K\},$$

where $K$ is the number of gesture classes. Since the size of assets required for classification is proportional to $O(K^2)$, for large $K$ the size of the classification database would grow prohibitively. Here we note that for most gesture classes the differences among them are large enough that the scoring function in Equation 5.3 is sufficient to disambiguate them; in training we identify “similar” gestures and train weighting matrices only for these sparse pairs. Of course, the density of required pairs in the complete $K \times K$ matrix depends on the similarity of the gesture motion classes.

Finally, we point to some interesting features of our classifier:

- The length of the input buffer does not have to equal the length of the class prototypes. Thus, shorter input sequences can be matched using the same algorithm. Only the normalization parameters of the cross correlation need to be adapted.

- Our algorithm returns as a side-effect the phase shift with respect to the prototype of the matched class. This information is useful to synchronize the user’s dancing pattern with the gaming platform.

- Errors reported by our system may often be benign, in particular for short input buffers. One characteristic of our classifier is that it returns in “near optimal” manner the best-matched class within the entire gesture database, as well as phase-shift within its prototype mean. Therefore, in scenarios where an avatar
renders the player’s motion, errors may pass unnoticed due to short-spanned cross-class similarities.

Moreover, the choice of the logistic regression with L2 regularization was made due to its simple training and evaluation procedures. Note that it is intimately related to linear SVMs and Ordinary Least Squares and on our dataset, many versions of these classifiers are expected to perform similarly.

5.4.5 Distance Measure

Once our classification method identifies the best matched class, the remaining question is how “well” the player performed this gesture compared to the oracle. Comparison with respect to the prototype mean (including the score obtained by correlation with it) is misleading as it outlines how well the player performed versus the average rather than the expert dancer. On the other hand, besides having a single scoring number, we favor a report that outlines how “well” the game player danced per joint. To resolve this problem, it is desired to obtain motion recordings labeled for artistic appeal, and to learn a regression model on this dataset that replicates the human expert. Even then, it is arguable how consistent human labeling is. To avoid the semantic nature of grading body motion, we measure the discrepancy between the relative motion of the current actor and the expert.

In our approach, we first globally align the feature sequence of the player using the phase-shift provided by the classification method. Subsequently, dynamic time warping is used to measure the discrepancy of the two signals considering the possible local misalignments. To overcome the outliers due to the noise we prefer a robust cost
at the computation of dynamic time warping, defined as:

\[
d(x, y) = \begin{cases} 
1 - e^{-(x-y)^4/\sigma^2}, & \text{if } |x - y| < \delta \\
1 - e^{-\delta^4/\sigma^2}, & \text{otherwise}
\end{cases}
\] (5.6)

where \(\sigma\) is a parameter that controls the amount of deviation from the expert’s performance allowed and \(\delta\) is a threshold minimizing the effect of outliers.

The proposed metric is parameterized to adjust to different motion accuracy standards along space and time by tightening and relaxing \(\sigma\) and \(\delta\). In our experiments, it behaved as a rather solid detector when computed against all oracles; still, its computational complexity was prohibitive for exploring per-class applications.

### 5.5 Experimental Results

**Benchmark.** The proposed technology has been tested against a large benchmark of 28 different gesture classes with a total of 11361 recordings. The cardinality of the training set is outlined in Table 5.2 with a majority of classes having more than 200 instances performed by at least a dozen and as high as 50 different subjects. All recordings were produced using Kinect’s STA at 30fps, to a variety of music clips and under different environmental settings. We linearly interpolated and normalized all recordings to 120 samples per 8 beats.

For each class, we also have an oracle recording; the skill level of all other participating subjects varied greatly. The complexity of the dance movements present in our dataset makes it appealing as the backbone of an interface for a videogame. In the same time, it is a challenging dataset for a classification algorithm, since many moves share “primitive” actions. Therefore, the temporal dynamics and the motion of small portion of the body are the only discriminative characteristics to achieve satisfactory classification performance.
Moreover, due to the sheer amount of data, the recordings were not labeled accurately, with gestures such as TouchOut and StepOut having a mislabeling rate of approximately 10%. In that respect, we ask the reader to treat the obtained experimental results as likely lower bounds on the results that could be achieved in the absence of mislabeled data.

\textbf{Performance.} We evaluate the proposed scheme on our dataset using 4-fold cross validation and report the average performance in Table 5.2. At each of the folds, the dataset was partitioned by randomly selecting dancers-subjects to approximately 70\% training and 30\% testing. Note that correlation of gestures coming from a single dancer is much higher than against any other dancer unless the ones considered are professionals. The classification performance for the entire 8 beat sequences \((N = 120 \text{ samples})\) is on average 96.9\%. The lowest performances observed are 81.1\% and 89.4\% for the TouchOut and StepOut gestures, respectively. As mentioned above, the main reason for those performances is the mislabeling occurred during their annotation. Also note here that, the classification accuracy for dance moves that contains body poses, which are extremely challenging for the Kinect’s STA to identify (e.g., KneeBend) is relatively lower due to the lower signal-to-noise ratio. Moreover, we evaluate the performance of the different components of our algorithm. Specifically, the classification performance is computed in the case of (a) using only the normalized cross-correlation features \((h_i(c_i), \text{Equation 5.3})\), (b) using only the energy features \((e_i(a_i), \text{Equation 5.3})\), (c) using only the first stage of the cascade classifier (class ranking, Section 5.4.3) and (d) using both two stages of classifier (logistic regression, Section 5.4.4). The results are summarized in Table 5.1.

We also tested our algorithm for shorter input sequences 2 and 4 beats, corresponding to 30 and 60 samples respectively. For the training of our algorithm we randomly sampled sub sequences of length 2 and 4 beats from the full 8 beat sequences of our
dataset. All the sub sequences where zero-padded to 8 beats length. The correlation distributions and the coefficients of the logistic regression were learned commonly for all the 2 and 4 beats subsequences, as described in Section 5.4. Interesting, the decrease of performance for classifying 4 beats sub sequences is on average only 0.8% compared to the 8 beats classifier performance. The ability of our scheme to accurately identify the human behavior among 28 highly complex dancing moves requiring only 2 seconds buffer (assuming that 1 music beat is 0.5 seconds) is of great importance for the usability of the propose interface. For the sub sequence of 2 beats length the average performance is 86.5%, actions that are composed by the same elementary moves are confused, e.g. ClapSlideNStomp and WhassupWalk.

Additionally, we developed a detector that considers shorter input buffers but with a known phase shift with respect to the prototype mean (Table 5.2). In this case, a search window of 11 frames was considered to identify the best max-likelihood score. The superior performance for the 4 beat sub sequences verifies the importance of the synchronization among the players with the prototype mean. Assuming an approximate knowledge of the phase alignment, the search for the best \( \tau \) can be restricted to a smaller range making the correlation statistics more discriminative.

Finally, we compare our algorithm with two efficient baseline methods for classification and retrieval of motion capture data [OFH08, SH08]. Onuma et al. [OFH08] proposed logarithmic scaled kinetic energy features and used a 1-nearest neighbor clas-
Table 5.2: Experimental results. \( L \) is the number of gesture recordings per class. The columns under \( 4 \times "70/30 split" \) illustrate the average classification accuracy using 4-fold cross validation. The columns under “Synched” show the average classification accuracy assuming an approximate knowledge of the phase alignment.

<table>
<thead>
<tr>
<th>Class</th>
<th>( L )</th>
<th>( 4 \times &quot;70/30 split&quot; )</th>
<th>Synched</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>8 beat</td>
<td>4 beat</td>
</tr>
<tr>
<td>2Step</td>
<td>714</td>
<td>98.2%</td>
<td>94.5%</td>
</tr>
<tr>
<td>ArmyWalk</td>
<td>181</td>
<td>94.7%</td>
<td>92.9%</td>
</tr>
<tr>
<td>BodyRoll</td>
<td>334</td>
<td>99.2%</td>
<td>95.0%</td>
</tr>
<tr>
<td>Bounce</td>
<td>200</td>
<td>95.5%</td>
<td>97.5%</td>
</tr>
<tr>
<td>ClapSlideNStomp</td>
<td>241</td>
<td>99.9%</td>
<td>99.7%</td>
</tr>
<tr>
<td>FunkyOldMan</td>
<td>391</td>
<td>99.8%</td>
<td>98.3%</td>
</tr>
<tr>
<td>Hip</td>
<td>259</td>
<td>99.2%</td>
<td>98.3%</td>
</tr>
<tr>
<td>KickSnap</td>
<td>286</td>
<td>99.2%</td>
<td>96.3%</td>
</tr>
<tr>
<td>KneeBend</td>
<td>186</td>
<td>95.0%</td>
<td>82.5%</td>
</tr>
<tr>
<td>LockinNew</td>
<td>231</td>
<td>95.8%</td>
<td>86.8%</td>
</tr>
<tr>
<td>MJHips</td>
<td>324</td>
<td>100.0%</td>
<td>95.0%</td>
</tr>
<tr>
<td>OpenClose</td>
<td>494</td>
<td>98.2%</td>
<td>98.3%</td>
</tr>
<tr>
<td>PointSwim</td>
<td>228</td>
<td>97.9%</td>
<td>97.7%</td>
</tr>
<tr>
<td>Roar</td>
<td>379</td>
<td>95.9%</td>
<td>94.5%</td>
</tr>
<tr>
<td>RockOn</td>
<td>448</td>
<td>99.8%</td>
<td>99.6%</td>
</tr>
<tr>
<td>SaluteStep</td>
<td>188</td>
<td>100.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>SideStep</td>
<td>626</td>
<td>96.6%</td>
<td>90.0%</td>
</tr>
<tr>
<td>SideStepHand</td>
<td>417</td>
<td>97.1%</td>
<td>97.5%</td>
</tr>
<tr>
<td>Skater</td>
<td>257</td>
<td>96.9%</td>
<td>97.1%</td>
</tr>
<tr>
<td>SlideNStomp</td>
<td>257</td>
<td>99.7%</td>
<td>99.0%</td>
</tr>
<tr>
<td>Step80</td>
<td>492</td>
<td>95.8%</td>
<td>91.6%</td>
</tr>
<tr>
<td>StepOut</td>
<td>847</td>
<td>89.4%</td>
<td>84.6%</td>
</tr>
<tr>
<td>StepTouch</td>
<td>293</td>
<td>96.3%</td>
<td>92.9%</td>
</tr>
<tr>
<td>StepTouchSnap</td>
<td>348</td>
<td>97.8%</td>
<td>97.0%</td>
</tr>
<tr>
<td>TheWave</td>
<td>1318</td>
<td>99.2%</td>
<td>94.6%</td>
</tr>
<tr>
<td>TopRock</td>
<td>425</td>
<td>97.1%</td>
<td>93.5%</td>
</tr>
<tr>
<td>TouchOut</td>
<td>787</td>
<td>81.5%</td>
<td>81.1%</td>
</tr>
<tr>
<td>WhassupWalk</td>
<td>496</td>
<td>97.7%</td>
<td>95.1%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>—</td>
<td><strong>96.9%</strong></td>
<td><strong>94.2%</strong></td>
</tr>
</tbody>
</table>
sifier based on their Euclidean distance. In order to implement the proposed method, we transformed the 3-D positions of skeletal joints of our dataset to Euler angle representation. The proposed distance function has been proven to be effective for classification of motion capture data performing “coarse” motions, e.g. “walking”, “running”, “boxing”. However, its average classification accuracy on our dataset is relatively low 13.76%, since the dance gestures of our dataset have subtle differences on their energy profiles. [SH08] developed a performance animation system that uses accelerometer measurements as input. Using the Haar wavelet transform on the acceleration signals they manage to obtain a real-time motion retrieval algorithm. The average classification accuracy obtained using the latter algorithm is 63.32%.

5.6 Conclusion

The emergence of affordable real-time depth sensors opens new avenues for interactivity. Compared to accurate optical or mechanical marker-based motion capture systems, depth sensors offer balance in usability and cost. The consequence is substantial increase in noise, in particular when we are faced with the fact that the world population is the user. The noise stems from the sensor, player’s physique, clothing, and kinetic IQ; the latter probably causing the most variability in particular in complicated motion patterns.

We have shown that in the context of dancing, a classifier can be designed and trained to recognize from among several dozen gestures in real-time and with high accuracy. The periodic nature of dance moves allowed us to specialize the classifier to efficiently detect and compensate for phase alignment. Scaling the classifier for \( K \) to be in the hundreds affects two system characteristics:

- **Computational workload** increases linearly with \( K \). At runtime, the critical
path in the system revolves around skeleton acquisition, gesture classification, and the system rendering player’s avatar on screen. Since sensor’s noise is substantial, rendering based on its output is highly undesirable, gesture classification with an intuitive distance metric is crucial in providing an immersive experience.

- **Error rates** in the system do not scale according to the Central Limit Theorem because the discriminatory noise is not i.i.d., it is well under the control of the game designer. To address this challenge, our system is designed to help the choreographer identify how discriminative a new gesture is, compared to an existing gesture database.

There are many areas for future work. Certainly, one could explore relaxing some of the assumptions of our classifier, such as the musical beat alignment, integrating raw depth sensor data to improve classification, providing user feedback as to quality of their motion to the point where the system could infer a gesture oracle from motion patterns of users with average kinetic IQ, etc. In addition to identifying gestures, it may be possible to learn interesting axes within the subspace of each gesture. Such axes might correspond to attributes such as emotion or energy. As an analogy, previous work in computer graphics has shown how to synthesize animation using “verbs and adverbs” [RCB02]. The goal of the reverse process would be to recognize not only the verb (gesture) but also its adverbs (attributes).
CHAPTER 6

Tracklet Descriptor -
Low Level Features for Video Analysis

6.1 Introduction

In this chapter, we define elementary spatio-temporal statistics in video sequences under a set of modeling assumptions about the image formation process (Section 6.2), propose a model to infer them (Section 6.2.2), and evaluate the resulting descriptors on classification tasks using benchmark video datasets (Section 6.4).

We focus on low-level representation, to devise statistics of the spatio-temporal signal that are insensitive to nuisance factors and yet sufficiently discriminative, that can be used as building blocks for more sophisticated models that exploit top-down structure and priors. Thus we purposefully operate with impoverished models that emphasize the low level, keeping top-down processing, shape and motion priors, and learning machinery to a minimum. Even with this impoverished representation, we show that we can achieve competitive performance in end-to-end classification tasks on benchmark datasets. More importantly, however, we believe that our features can be profitably used by more sophisticated models that do exploit top-down information in the form of global temporal statistics or spatial context.

We propose spatio-temporal feature descriptors that capture the local structure of the image around trajectories tracked over time. We actively restrict our attention to a
subset of the spatial image domain and encode its “local photometry”. Our approach differs from “holistic” (Section 2.1) [BD01, EBM03a, ZI06, LJD09, YW09] that use the entire video volume to extract global statistics, and compare them with standard norms, block correlation [ZI06], or dynamic time warping [LJD09]. Unlike these approaches, we explicitly model “simple” nuisance variability (position, contrast etc.), detect a corresponding frame with a co-variant detector, and “undo” it in the descriptor, which is therefore by construction invariant to such nuisances. The residual “complex” nuisances (local deformation, deviation from Lambertian reflection, complex illumination changes) are instead averaged out in the descriptor. Such averaging is performed relative to the structure of the nuisances, learned during the training phase, and plays a similar role to spatial binning (a form of “unstructured” averaging) in [Low99]. In this sense, our approach relates to local features based representations (Section 2.2) for action recognition, including [SLC04, DRC05b, LLS09, NBT07a, WUK09]. Different local descriptors have been proposed to capture shape [SLC04, DRC05b] or joint motion and shape [LP07, LMS08, CMP10] by aggregating features within video cubes centered at spatio-temporal interest points into a static descriptor. In contrast, we retain in our tracklet descriptor the entire feature time series from birth to death of each tracked region.

We illustrate the general architecture of our descriptors using off-the-shelf detectors and local motion estimators and perform averaging or aggregation using the computational architecture of [Low99]. While more sophisticated instantiations are possible, already these simple choices attain state-of-the-art performance in the Activities of Daily Living (ADL) [MPK09], the KTH [SLC04] and the Hollywood Human Action (HOHA) [LMS08] datasets. The implementation of the proposed descriptor is available at: http://vision.ucla.edu/~raptis/tracklets.
6.2 Spatio-temporal Tracklet Descriptors

We now describe the modeling assumptions under which we operate, and the procedure to infer the resulting representation (Section 6.2.2). While one would want to assemble these elementary actions (dictionary elements) into a model that captures the joint spatio-temporal statistics at a more global spatial scale (“context”), in Section 6.4 we show that even a naive use of the dictionary labels as a “spatial bag” yields competitive performance in end-to-end tasks.

6.2.1 Model and assumptions

We assume that each “object” is defined at rest as a compact region of space, only part of which may be visible due to occlusions, and projected onto a subset \( D \) of the image plane, yielding a function \( \rho : D \subset \mathbb{R}^2 \rightarrow \mathbb{R}^+; \ x \mapsto \rho(x) \) where \( D \subset \Omega \) is the base image region. There is no requirement that an entire object be captured by one base region. Instead, we can expect objects to be over-segmented in multiple base regions, with their spatio-temporal relations characterizing the object.\(^1\) Base regions move under the action of a finite-dimensional group \( g(t) \in G \), which we assume without loss of generality to be the group of rigid motions \( G = SE(2) \), with the residual motion, that depends on the shape of the scene and viewpoint, captured by a general diffeomorphism \( w : \Omega \rightarrow \Omega; \ x \mapsto w(x) \). Finally, a contrast transformation is applied to the range of the image in the base region, and all other photometric factors (specularities, translucency, inter-reflections etc.) are lumped together as an additive

\(^1\)Although it is precisely these contextual spatial relations that we ignore in Section 6.4, to test the representational power of the descriptor alone.
component \( n(x,t) \). These assumptions are summarized in the model:

\[
\begin{aligned}
\rho(x), \ x \in D \subset \mathbb{R}^2 & \quad \text{base region} \\
\rho \circ g(t) = \rho(g(t)x), \ g(t) \in SE(2) & \quad \text{global motion} \\
\rho \circ w(x,t) \circ g = \rho(w(g(t)x,t)) & \quad w : \mathbb{R}^2 \to \mathbb{R}^2 \quad \text{local deformation} \\
h(t) \circ \rho \circ w \circ g = h(\rho(w(g(t)x,t))), t) & \quad h : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+ \quad \text{contrast} \\
I(x,t) = (h \circ \rho \circ w \circ g)(x,t) + n(x,t) & \quad \text{complex illumination, noise, etc.}
\end{aligned}
\]

(6.1)

The above equation is valid only for those \( x \in \mathbb{R}^2 \) that intersect the domain of the image \( \Omega \). Elsewhere, the image is due to phenomena other than the base region, which we call clutter, \( \beta(x,t) \). So, the actual measured image is given by

\[
I(x,t) = \begin{cases} 
  h \circ \rho \circ w \circ g(x,t) + n(x,t), & \forall \ x \in g^{-1}(D)w^{-1}(D,t) \cap \Omega \\
  \beta(x,t) & \text{elsewhere.}
\end{cases}
\]

(6.2)

The components (hidden factors) of the extended temporal observation of an object are the (multiple) base image regions \( \rho_i \mid D \), their (variable) length \( \hat{T}_i = T_i - \tau_i \), global trajectory \( \{g_i(t)\}_{t=\tau_i}^{T_i} \), their local deformation\(^2\) \( \{w_i(x,t) ; \ x \in g_i(t)D\}_{t=\tau_i}^{T_i} \), the contrast transformation \( \{h_i(t)\}_{t=\tau_i}^{T_i} \), while everything else is lumped in \( n_i(x,t) \). In the rest of this section we will omit the index \( i \) and focus on inference and representation: How can we “extract” the hidden components from a time series \( \{I(x,t), x \in \Omega\}_{t=\tau}^{T} \)? What components of the data-formation process matter for classification? In order to make the inference tractable, we make the following modeling assumptions: The effect of complex nuisances \( n(x,t) \) is small relative to other factors, so we (a) seek explanations of the data that minimize their effects (e.g. a suitable norm of \( n(x,t) \)). The contrast transformation \( h \) “contains no information” (i.e., we wish the outcome of the task to be independent of contrast), so we (b) seek to eliminate it from the representation. The

\(^2\) Here \( g_iD = \{g_i(x) \mid x \in D\} \).
global motion \( g(t) \) may or may not contain information, depending on the task, so we (c) seek to infer it from the data for later use, or to (d) provide a local reference where to compute the deformation field \( w(x,t) \). The base region \( \rho \) and the local deformation \( w \) contain all the photometric, geometric and dynamic information, respectively, embedded in the data. Therefore, the inference problem can be stated as:

\[
\{ \hat{\rho}, \hat{w}, \hat{g} \} \subset T = \arg\min_{\rho, w, D, h, g} \int_\tau^T \|n(x,t)\|_D dt \tag{6.3}
\]

subject to (6.2), where \( \|n(x,t)\|_D = \int_D |n(x,t)|^2 dx \), with the addition of an area regularizer to avoid the trivial solution \( D = \emptyset \). This formalizes (a). To eliminate \( h \), (b) we simply encode the estimate of the base image region \( \hat{\rho}I(x) \hat{\rho} = I \circ \hat{w}^{-1} \circ \hat{g}^{-1} \) using a complete contrast-invariant, such as the geometry of the level lines (or its dual, the gradient orientation), or a local contrast normalization, e.g.

\[
\phi(\hat{\rho}(x)) = \frac{\nabla \hat{\rho}I(x)}{\|\nabla \hat{\rho}I(x)\|_\epsilon} \quad \text{or} \quad \phi(\hat{\rho}(x)) = \frac{I - \int_D I dx}{\|\text{std}(I_{|D})\|_\epsilon} \tag{6.4}
\]

where \({\|I\|}_\epsilon = \min\{\|I\|, \epsilon\}\). We are then left with estimating (c) the global motion \( g \), and (d) the local deformation \( w \). Rewriting Equation 6.3 we have a sequence of equivalent optimizations in fewer and fewer unknowns:

\[
\arg\min_{h, \rho, w, g} \int_\tau^T \int_D |I(x,t) - h \circ \rho \circ w \circ g| dx dt = \text{ (thm. 7.4, p. 269 of [Rob01])}
\]

\[
= \arg\min_{\rho, w, g} \int_\tau^T \int_D |\phi(I(x,t)) - \phi(\rho \circ w \circ g)| dx dt = \text{ (thm. 1, p. 4 of [SY02])}
\]

\[
= \arg\min_{w, g} \int_\tau^T \int_D |\phi(I(x,t)) - \phi(I(x,t+1) \circ w \circ g)| dx dt = \{ \hat{g}(t), \hat{w}(x,t) \} \tag{6.5}
\]

This problem can be solved using variational optimization techniques [SY02]; a more efficient, albeit suboptimal, solution can be arrived at by first assuming \( w(x,t) = x \)

---

3Since the gradient direction will be weighted by its norm in the averaging operation to compute the descriptor (Section 6.2.2), the value of \( \epsilon \) does not matter in practice. As an alternative, when color images are available, one can use spectral ratios or local normalization to eliminate contrast transformations.
and estimating \( \hat{g}(t) = \arg\min_g \int_T \int_D |\phi(I(x,t)) - \phi(I(x,t+1) \circ g(t))| \, dx \, dt \) with any tracking algorithm [LK81, ST94, MPK09]. Then, given \( \{ \hat{g}(t) \}_{t=1}^T \), estimate \( \hat{w}(x,t) = \arg\min_w \int_T \int_D |\phi(I(x,t)) - \phi(I(x,t+1) \circ w \circ \hat{g}(t))| \, dx \, dt \) with any optical flow algorithm. Note that \( \hat{w} \) depends on \( \hat{g} \), and there is no guarantee that substituting \( \hat{w}, \hat{g} \) in (6.5) minimizes the cost. However, this approach is sufficient for our purposes, otherwise one can revert to an infinite-dimensional optimization of (6.5).

### 6.2.2 Simplest instantiation and inference of the representation

Following the derivation above, given a video sequence \( \{ I(x,t), \, x \in \Omega \}_{t=1}^T \), we first select candidate regions via any feature detector [Low99, HS88, BTV06], and track them over time using a contrast-compensated translational tracker to obtain a number of trajectories \( \{ \hat{g}_i(t) \}_{t=\tau_i}^{T_i} \) of varying length \( \hat{T}_i \), addressing (c). Many trackers also provide a rotational and scale reference; the latter can be used to select the base regions \( D_i \subset \mathbb{R}^2 \). The former can be used to fix local orientation, although we select the vertical image coordinate as reference. In the resulting local frame \( \{ D_i, \hat{g}_i(t) \} \) we then estimate the local motion \( \{ \hat{w}_i(x,t) \}_{t=\tau_i}^{T_i} \) using any of a number of local optical flow algorithms, the simplest being [LK81]. This addresses (d) and completes the (co-variant) frame selection process. Therefore, we design an invariant descriptor by representing the image in the selected frame, \( \{ D_i, \hat{g}_i(t) \} \) via the contrast invariant \( \{ \phi(I \circ \hat{g}_i) \} \), and concatenate that with the motion field \( \{ \hat{w}_i(x,t) \circ \hat{g}_i(t) \} \) in the base region \( D_i \).

If we had priors on the intra-class variability \( dP(g, w) \), we would marginalize the resulting descriptor; in their absence, it is common to assume that the object or category of interest is described by an “uncertainty ball” around a reference descriptor, that is therefore “blurred” in some sense, ideally by averaging with respect to the prior, but more often by coarse spatial binning. In the latter case, the descriptor for \( \{ \phi(I \circ \hat{g}_i) \} \)
corresponds to a histogram of gradient orientations (HoG) [Low99, DT05], and the descriptor for $\{\hat{w}_i(x, t) \circ \hat{g}_i(t)\}_{D_i}$ corresponds to a histogram of optical flow vectors (HoF).

Although many have used HoG/HoF descriptors [LP07, LS08, LMS08, CMP10], they aggregate them into a static signature, whereas our previous analysis and [Soa10] suggest retaining their temporal evolution. However, rather than averaging by spatial binning (that presumes ergodicity), we prefer to use at least a crude approximation of the prior $dP(g, w)$ in the form of samples $\{g(t_j)\}$, $\{w(x, t_j)\}$ inferred during the training phase. The resulting descriptor, which we call AoG (average of gradient orientation) and AoF (average of optical flow), averages over the training samples – inferred in a sliding temporal window $\{t_j\}_{j=1}^L$ and thought of as samples from an importance distribution:

$$AoG(t|x, g_i, D_i) = \sum_{\tau = t-\lfloor L/2 \rfloor}^{t+\lfloor L/2 \rfloor} \phi(I(x, \tau)) \circ g^{-1}_i(\tau) \quad x \in g_i(\tau) D_i \cap \Omega$$

where $g_i D_i$ is defined in footnote 2. Although “oG” in AoG stands for the gradient orientation, in analogy to HoG, any other contrast-normalizing statistic $\phi$ can be used, as in (6.4). Similarly, we have

$$AoF(t|x, g_i, D_i) = \sum_{\tau = t-\lfloor L/2 \rfloor}^{t+\lfloor L/2 \rfloor} (w_i \circ g_i)(x, \tau) \quad x \in D_i \cap \Omega$$

We call Tracklet Descriptor (TD) the concatenation of the entire time series of either HoG/HoF, or AoG-HoF, and compare the two in Section 6.4, where we show the latter to yield marginally improved performance at a significantly lower computational cost. Optionally, the TD can be augmented with some sample statistic, for instance the trajectory relative to the spatial or spatio-temporal mean.

$$\pi_i(t|I) \doteq \{A/HoG_i(t), A/HoF_i(t)\}$$

89
As stated in Section 6.1, we postulate *compositionality* of our representation, so it is natural to organize tracklet descriptors into a “dictionary.” However, because we retain the entire time series, the process is more involved as descriptors of different length have to be compared. In Section 6.3 we describe how this can be done using dynamic time warping and clustering by affinity propagation. As an alternative to averaging, one could consider histograms aggregated over time, rather than space, with similar results, as advocated by [LS10].

### 6.3 Implementation

Following Section 6.2.2, we reduce the group \( G = \mathbb{R}^2 \) to pure translations, and estimate \( \{ \hat{g}_i(t) \in G \}_{t=\tau_i}^{T_i} \) using [ST94], as implemented by [Bir96], without affine consistency check, similar to [MPK09]. Features lost during tracking are replaced by newly selected ones. We prune tracks that are less than \( T_i = 5 \)-frames long, or that move less than \( \hat{g}_i(T_i) = 3 \)-pixels in standard deviation. Unlike [SWY09], we do not impose an upper bound on \( \hat{T}_i \), and unlike [DRC05b, SLC04, CMP10, MHS09] we do not use a fixed time-scale.

#### 6.3.1 Constructing tracklet descriptors

We capture the contrast-invariant statistics \( \phi \) of the base regions \( D_i \) using the gradient orientation spatially binned (HoG) or averaged (AoG) in a sliding temporal window, e.g., \( L = 5 \) with fixed scale and orientation, centered at each spatial location \( \hat{g}_i(t) \) along the trajectory. The size of \( D_i(t) \) could be adapted using the scale component estimated on-line by the tracker. Although we estimate rotation of the base regions \( D_i \) we discard it, and use the vertical component of the image plane as a reference. In yet a simpler instantiation, one can consider the base regions \( D_i \) fixed to, say, 18 × 18 or 32 × 32 pixels. We estimate the local deformation \( \hat{w}_i(x, t) \) using [LK81] and aggregate...
it either in a spatial histogram (HoF) or in an average (AoF) within each region $D_i$. While HoG/HoF result in a fixed 128-dimensional vector each, AoG/AoF have variable size depending on $|D_i|$; therefore they are quantized into a comparable number of components (225 in the experiments, corresponding to $15 \times 15$ patches). The two vectors are concatenated$^4$ and stacked sequentially over time into a matrix.

### 6.3.2 Tracklet dictionary

For each base image region $D_i$, a tracklet descriptor represents a multi-dimensional time series, $\pi_i : [\tau_i, T_i] \rightarrow \mathbb{R}^N$. To define a distance between two descriptors we must discount initial time, speed of execution, and duration of an action. Therefore, we adopt the dynamic time warping (DTW) distance [SC78]:

$$d(\pi_i, \pi_j) = \inf_{\alpha, \beta \in \mathcal{H}} \frac{1}{M} \sum_{t=1}^{M} \| \pi_i(\alpha(t)) - \pi_j(\beta(t)) \|_1$$

(6.9)

where $\alpha, \beta \in \mathcal{H}$ are continuous monotonic transformations [VCR06, LJD09] of the temporal domain. For HoG, HoF and AoG we use the $\ell_1$ distance. Optical flow vectors, however, are not sparse, so $\ell_2$ should be used instead, allowing small discrepancies. Therefore, AoG and AoF cannot be simply concatenated, but instead separate dictionaries, and combinations of separate kernels, have to be learned. The different structures of AoG and HoF also do not lead to a “meaningful” compact descriptor. To make comparison as fair as possible, in Section 6.4 we test AoG vs. HoG in isolation (Table 6.3). For a track of 100 frames, HoG takes 13 seconds to be computed (in non-optimized C code), whereas AoG takes 0.6 seconds (in Matlab).

Because of the variable length, many commonly used clustering algorithms (e.g., k-means) are inapplicable to clustering time series. Agglomerative clustering [KRW02] and k-medoids have been used to select cluster centers for time series. We compute

---

$^4$Although one could introduce weights between the spatial and temporal component, and optimize the weight to a particular dataset, we do not do so in Section 6.4.
pairwise distances among tracklet descriptors, and set the distance to infinity for pairs with length ratio not between 0.5 and 2, since DTW does not provide a meaningful warping path for those cases [RJ93]. We use affinity propagation [FD07] to cluster and select dictionary elements. This method is efficient due to the sparsity of the initial distance matrix and effective to define discriminative exemplars without the need of multiple random initializations that algorithms like k-centers require. In our experiments the size of the dictionaries was not pre-specified but it was automatically selected by affinity propagation.

It is not immediate to visualize our cluster centers, since our model is not strictly generative. However, Fig. 6.1 shows parts of the tracks colored according to their nearest neighbor in a tracklet dictionary. Fig. 6.2 shows a sample trajectory with samples of the quantized histogram of gradient orientations and optical flow super-imposed on the image. These histograms are concatenated to form a temporal sample of the time series \( \{\pi_i(t)\} \).

### 6.3.3 A basic classification scheme

The simplest recognition method we consider is akin to a bag-of-features [CDD04], whereby we discard global temporal ordering, capturing only the local temporal variation of a tracklet. This admittedly naive model achieves performance already close to the state of the art. Given a codebook of TDs, we assign each trajectory in a test frame to the closest codebook element (Section 6.3.2); then each video is represented by a histogram of occurrences of dictionary elements. We use a support-vector machine with either a RBF-\(\chi^2\) kernel or an intersection kernel. The penalty parameter is selected by 10-fold cross-validation in the training set, whereas the scale parameter of the RBF kernel is selected as the mean \(\chi^2\) distance of the training samples. The RBF-\(\chi^2\) SVM achieves an improvement of \(1 - 2\%\) over the intersection one.
Figure 6.1: Tracks extracted from ADL, KTH and HOHA datasets. Color indicates their label based to the tracklet descriptor dictionary.
Figure 6.2: A track with samples of the histogram of gradient orientation (left, blue) and histogram of optical flow (right, red) along the trajectory. These are concatenated to form a 256-dimensional temporal sample of the time series that represents that elementary action.

6.4 Experimental Evaluation

We evaluate the proposed scheme on three publicly available datasets: KTH [SLC04] Activities of Daily Living (ADL) [MPK09] and Hollywood Human Actions (HOHA) [LMS08]. As pointed out in Section 6.3.2, AoG cannot be simply concatenated with either AoF or HoF, but has to be combined using multiple kernels. In our first two experiments we use the compact tracklet descriptor based on the HoG/HoF, so we can use one dictionary and one kernel, and have a fair comparison with existing local descriptors [DRC05b]. In the most challenging dataset (HOHA) the individual components HoG and AoG are compared in Table 6.3, and their combination with HoF is reported in Table 6.4.

KTH is chosen because of its popularity, though its modest spatial (160 × 120 pixels) and temporal (25 frames per second) resolution make for an impoverished data
stream that is not well suited for local representations. There are 6 actions performed by 25 subjects in 4 scenarios (outdoors (s1), outdoors with scale variation (s2), outdoors with different clothes (s3) and indoor (s4)), resulting in 598 clips. The simplicity of these actions, combined with an uncluttered static background, make this dataset ideally suited for global representations [LJD09]. Nevertheless, even without exploiting background subtraction or the global evolution of the silhouette (hard to obtain in most realistic scenarios), our scheme is competitive with the state of the art (Table 6.1).

More specifically, we track an average of 340 trajectories per video with an average length $\hat{T}_i = 23$ frames. Low resolution and the compression artifacts are a challenge to tracking, so the average length is relatively small. Our base regions $D_i$ are fixed at $18 \times 18$ pixels, similar to the spatial size of Cuboids [DRC05b, NWF08, NBT07a]. Examples of tracks and the corresponding HoG descriptors are shown in Fig. 6.3. The classification performance of algorithms that use spatio-temporal descriptors computed in volumes around interest points [LLS09, LMS08, CMP10, NBT07a, DRC05b, NWF08] has proven that the choice of the temporal scale is crucial. Laptev et al. [LMS08] construct static HoG/HoF around points detected by spatio-temporal Harris-3D [Lap05] at multiple scales, using $\Delta t = 25, 36$; [CMP10] computes a HoG/HoF around points detected by [Low99] in a volume with $\Delta t = 60$. Instead, our descriptors have variable temporal length depending on the image region $D_i$. Moreover, the optical flow in the image regions $D_i$ can be estimated reliably. This is not the case for the spatio-temporal cubes around a specific interest point.

We use leave-one(person)-out cross validation and average the result over the 25 permutations. To construct the codebook we use a relatively small training set, similar to [NWF08], to examine the generalization of our algorithm. We only use the descriptors extracted from the first two parts of the 72 videos of 3 subjects. Those descriptors
are excluded from the test and training sets. It should be noted that [LLS09, CMP10] used the videos of 24 subjects to construct the codebook, whereas [LMS08] used 8 subjects. Using a codebook with 1560 TDs of HoG/HoF, we achieve 94.5% recognition rate using RBF-χ^2 SVM (Table 6.1) considering the dataset as a single large set (all average in one). Using linear SVM with intersection kernel we achieve 93.82% recognition rate. Considering each scenario separately the recognition rate is: (s1) 98%, (s2) 92.67%, (s3) 91.95%, (s4) 96.67%.

Figure 6.3: Example of the tracks and an instance of the corresponding appearance descriptor of a boxing action on the KTH dataset.

The ADL dataset has higher-resolution (1280 × 720 pixels at 30FPS) with 10 different complex activities targeted to an assisted living scenario (e.g. “answering phone (aP),” “eating snack (eS),” “eating banana (eB)”). Five subjects perform each activity thrice for a total of 150 clips of duration varying between 10 and 60 seconds. It has drawbacks similar to KTH, in that all actions are taken against a still background from a fixed vantage point, an incentive to overfitting by using background subtraction and global statistics such as the absolute position of tracks in the image. Despite not using absolute positions, a simple classifier based on TDs HoG/HoF outperforms the state
Table 6.1: Performance comparison on KTH dataset.

<table>
<thead>
<tr>
<th>Our tracklets</th>
<th>evaluation</th>
<th>Recognition Rate</th>
<th>Structural Information</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>all scenarios in one</td>
<td>average of all scenarios</td>
</tr>
<tr>
<td>Niebles et al. [NWF08]</td>
<td>Leave-One-Out</td>
<td>94.5%</td>
<td>94.8%</td>
</tr>
<tr>
<td>Dollár et al. [DRC05b]</td>
<td>Leave-One-Out</td>
<td>81.5%</td>
<td>N/A</td>
</tr>
<tr>
<td>Schuldt et al. [SLC04]</td>
<td>Split</td>
<td>71.7%</td>
<td>N/A</td>
</tr>
<tr>
<td>Nowozin et al. [NBT07a]</td>
<td>Split</td>
<td>84.7%</td>
<td>N/A</td>
</tr>
<tr>
<td>Liu et al. [LS08]</td>
<td>Leave-One-Out</td>
<td>N/A</td>
<td>94.15%</td>
</tr>
<tr>
<td>Lin et al. [LJD09]</td>
<td>Leave-One-Out</td>
<td>93.4%</td>
<td>95.8%</td>
</tr>
<tr>
<td>Messing et al. [MPK09]</td>
<td>Split</td>
<td>74%</td>
<td>N/A</td>
</tr>
<tr>
<td>Yao et al. [YZ09]</td>
<td>Split</td>
<td>87.8%</td>
<td>N/A</td>
</tr>
<tr>
<td>Laptev et al. [LMS08]</td>
<td>Split</td>
<td>91.8%</td>
<td>N/A</td>
</tr>
<tr>
<td>Jhuang et al. [JSW07]</td>
<td>Split</td>
<td>N/A</td>
<td>91.7%</td>
</tr>
<tr>
<td>Schindler et al. [SV08]</td>
<td>Split</td>
<td>92.7%</td>
<td>90.7%</td>
</tr>
<tr>
<td>Yeffet et al. [YW09]</td>
<td>Split</td>
<td>90.1%</td>
<td>N/A</td>
</tr>
<tr>
<td>Chen et al. [CMP10]</td>
<td>Leave-One-Out</td>
<td>95.0%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

of the art by a sizeable margin. We extract on average 1300 tracks with mean duration \( \bar{T}_i = 110 \) frames. The base regions \( D_i \) are fixed at \( 36 \times 36 \) pixels. We again use leave-one (person)-out evaluation, similar to [MPK09, MP09], and report the average over the 5 permutations of the dataset. We randomly sampled \( 25K \) tracklets from the training set and constructed a dictionary with 2900 elements. Using this dictionary we achieve 82.67% average recognition rate using RBF-\( \chi^2 \) SVM (Table 6.2). Comparison to [MPK09] shows that our tracklet descriptor achieves comparable results without using any structural information (relative position or absolute position). It outperforms [MPK09] even when their classifier uses the position of the extracted trajectories relative to the position of the face of the actor. In order to have a fair comparison with existing methods that report results in the ADL dataset, we incorporate a codebook of the absolute position \((\bar{g}_i(t), \bar{t})\) of the tracks with size 60 obtained using K-means. Combining linearly the two \( \chi^2 \) kernels, we achieved 90% average recognition rate. We
Table 6.2: Performance comparison on ADL. Despite not using structural information or background subtraction, we improve the state of the art by a large margin. Using structural information, which we do not advocate, we can further improve recognition rate to 90%, highlighting the limitations of this particular dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Tracklets</td>
<td>82.67%</td>
</tr>
<tr>
<td>Spatio-temporal cuboids [DRC05b] (implemented by [MP09])</td>
<td>43%</td>
</tr>
<tr>
<td>Velocity Histories [MPK09]</td>
<td>63%</td>
</tr>
<tr>
<td>Latent Velocity Histories [MPK09]</td>
<td>67%</td>
</tr>
<tr>
<td>Augmented Velocity Histories with Relative Position [MP09]</td>
<td>72%</td>
</tr>
<tr>
<td>Augmented Velocity Histories with Relative and Absolute Position [MPK09]</td>
<td>89%</td>
</tr>
</tbody>
</table>

should note that, although absolute position is relevant in this dataset, and in particular it helps boost the performance of our algorithm as well as [MPK09] significantly, it does so only because all sequences are taken from the same vantage point, in an environment with fixed layout. In general, we advocate not using absolute position, even if it improves the performance in this particular dataset.

The **HOHA** dataset overcomes the limitations of ADL and KTH. The dataset contains 430 movie videos (240 × 450 at 24FPS) with challenging camera motion, rapid scene changes and cluttered and unconstrained background. Moreover, the human actions that are included are not constrained to single actor behaviors, e.g. “Sit down”, but also interactions between humans, e.g. “Kiss”, and objects, e.g. “Get Out of a Car”. We evaluate our trajectory descriptors following the experimental setting proposed by [LMS08], i.e. the test set has 211 videos with 217 labels and the training set has 219 videos with 231 labels (manually annotated). For each action we train a binary classifier and we evaluate our performance with average precision (AP) of the
precision/recall curve.

In order to manage the large variability of the image sequences contained in the dataset, features [ST94] are detected in multiple scales. We extract on average 500 tracks with mean duration $\hat{T}_i = 51$ frames. For each image region $D_i$ a HoG, HoF and AoG descriptor is constructed as described in (Section 6.3). First, a dictionary is created for each individual component of our tracklet descriptors and we evaluate its performance using RBF-$\chi^2$ SVM (Table 6.3). Our TD of optical flow significantly outperforms the HoF proposed by Laptev et al. [LMS08], proving to be more robust to background motion and large viewpoint changes. We also note that the performance of TD HoF is slightly worse than the trajectory transition descriptor (TTD) [SWY09], which is combined with spatio-temporal grid to incorporate some structural information in the descriptor. Our TD of AoG outperforms marginally both our TD HoG and the HoG of [LMS08], at a significantly reduced computational cost. Next, we construct our compact HoG/HoF tracklet descriptor and with a codebook with 2220 elements we achieve 32.1% mean average precision (MAP) (Table 6.4).

In order to fuse the TD AoG feature descriptor with TD HoF feature in our classification framework, we build a kernel as a convex combination of their $\chi^2$ kernels: $K_{AoG-HoG} = \lambda K_{AoG} + (1 - \lambda)K_{HoF}$, $\lambda$ was selected using cross-validation in the training set. The performance of the obtained kernel is 34.3% MAP. Our TD descriptors outperforms all the local descriptors that have been evaluated in HOHA dataset in a bag-of-features setting [KMS08, MHS09, LMS08] and we are competitive with the holistic approach proposed by [YW09] and the methods that use multi-channel Gaussian kernels [LMS08, SWY09] for combining the 48 or more channels provided by spatio-temporal grids.
Table 6.3: Performance comparison on HOHA Dataset of Individual components of Descriptors.

<table>
<thead>
<tr>
<th>Class</th>
<th>Our Tracklet</th>
<th>Laptev et al. [LMS08]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HoG / HoF</td>
<td>AoG / HoF</td>
</tr>
<tr>
<td></td>
<td>BoF</td>
<td>BoF</td>
</tr>
<tr>
<td>Answer phone</td>
<td>24.9%</td>
<td>22.1%</td>
</tr>
<tr>
<td>Get out of car</td>
<td>21.1%</td>
<td>19.3%</td>
</tr>
<tr>
<td>Hand shake</td>
<td>20.4%</td>
<td>19.1%</td>
</tr>
<tr>
<td>Hand shake</td>
<td>22.3%</td>
<td>28.2%</td>
</tr>
<tr>
<td>Kiss</td>
<td>48.4%</td>
<td>47.0%</td>
</tr>
<tr>
<td>Sit down</td>
<td>21.8%</td>
<td>22.2%</td>
</tr>
<tr>
<td>Sit up</td>
<td>16.7%</td>
<td>17.5%</td>
</tr>
<tr>
<td>Stand up</td>
<td>40.5%</td>
<td>59.9%</td>
</tr>
<tr>
<td>MAP</td>
<td>27.1%</td>
<td>29.4%</td>
</tr>
</tbody>
</table>

Table 6.4: Performance comparison on HOHA Dataset.

<table>
<thead>
<tr>
<th>Class</th>
<th>Our Tracklet</th>
<th>Laptev et al. [LMS08]</th>
<th>Yeffet et al. [YW09]</th>
<th>Matikainen et al. [MHS09]</th>
<th>Kläser et al. [KMS08]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HoG / HoF</td>
<td>AoG / HoF</td>
<td>HoG / HoF</td>
<td>HoG / HoF</td>
<td>HoG / HoF</td>
</tr>
<tr>
<td></td>
<td>BoF</td>
<td>BoF</td>
<td>BoF</td>
<td>BoF</td>
<td>BoF</td>
</tr>
<tr>
<td>Answer phone</td>
<td>26.7%</td>
<td>33.0%</td>
<td>26.7%</td>
<td>32.1%</td>
<td>35.1%</td>
</tr>
<tr>
<td>Get out of car</td>
<td>28.1%</td>
<td>27.0%</td>
<td>22.5%</td>
<td>41.5%</td>
<td>32.0%</td>
</tr>
<tr>
<td>Hand shake</td>
<td>18.9%</td>
<td>20.1%</td>
<td>23.7%</td>
<td>32.3%</td>
<td>33.8%</td>
</tr>
<tr>
<td>Kiss</td>
<td>51.5%</td>
<td>53.7%</td>
<td>52.0%</td>
<td>53.3%</td>
<td>57.6%</td>
</tr>
<tr>
<td>Sit down</td>
<td>23.8%</td>
<td>27.4%</td>
<td>37.8%</td>
<td>38.6%</td>
<td>36.2%</td>
</tr>
<tr>
<td>Sit up</td>
<td>23.9%</td>
<td>19.0%</td>
<td>15.2%</td>
<td>18.2%</td>
<td>13.1%</td>
</tr>
<tr>
<td>Stand up</td>
<td>59.1%</td>
<td>60.0%</td>
<td>45.4%</td>
<td>50.5%</td>
<td>58.3%</td>
</tr>
<tr>
<td>MAP</td>
<td>32.1%</td>
<td>34.3%</td>
<td>32.9%</td>
<td>38.4%</td>
<td>36.8%</td>
</tr>
</tbody>
</table>
6.5 Discussion

In this chapter, we have presented local spatio-temporal descriptors intended as low-level statistics to be used in action recognition systems. Our descriptors are deduced from an explicit model with all assumptions explicitly stated. They do not involve top-down modeling and can be efficiently learned from data. They can capture the discriminative statistics of the local causal structure of the data (temporal ordering), and the local shape and deformation of each base region. However, they do not encode neither the spatial structure nor the temporal constraints of the action. We introduce a more complex model for the recognition and classification of actions using our tracklet descriptor as a building block in Chapter 7.
Figure 6.4: Confusion matrices for ADL dataset (Top) and for KTH dataset (Bottom).
CHAPTER 7

Discriminative Discovery of Action Parts
from Mid-Level Video Representations

7.1 Introduction

We develop a mid-level representation of video data that can be inferred from video sequences for the purpose of classifying human action. In this chapter we consider both the recognition (what action is being performed) and localization (which portions of the video represent the action) tasks. Our approach brings together two lines of work: first, the extraction of statistics from sparse spatio-temporal descriptors, which as Chapter 6 showed contains sufficient information to perform action classification, while discarding a vast portion of the input video. Second, the rich structured models for classification in object recognition [WWP00, FGM10] that capture constraints among the parts of an object (an action in our case) in terms of an energy function and cast the recognition and localization problem in terms of explicit optimization over the part locations.

The main contribution of this chapter is to develop a spatio-temporal latent action model that is able to discriminatively identify salient structures and their pairwise relationships. Specifically, the low-level representation proposed in Chapter 6 is used to describe the spatio-temporal variability of a collection of moving frames, locally in space and time. These are grouped based on spatial and dynamic similarity, and consti-
tute putative “parts” in a complex spatio-temporal event, action or activity portrayed in a video snippet. The association of “groups” to “parts” is performed discriminatively by employing latent variables. During the training phase, the classifier parameters are learned simultaneously with the group association using a weakly annotated training set.

Our model takes as input this video representation in terms of clusters of trajectories and determines which among them should be used to ‘instantiate’ the parts of an action class. The quality of a given instantiation is phrased in terms of an energy of a Markov random field (MRF), which allows us to use the machinery of MRF optimization to perform the classification task. The model can thus be used both to classify an entire video snippet, but also for the purpose of analysis, to highlight local associations and determine which parts are relevant to a given classification task. The learning of the cost function that drives the matching is performed discriminatively using large margin learning of a ranking function, while the individual part properties and the relations among them are estimated in a bootstrapped, concave-convex procedure [YR03].

We test our approach using the Hollywood Human Action (HOHA) [LMS08] dataset (Section 6.4), where it performs competitively in the overall classification task, while having the added benefit of enabling local analysis. In a sense our method can support more fine-grained decision tasks than reflected in the available benchmark datasets, including for instance the discovery of spatio-temporal relations between parts and the ‘parsing’ of a video into action parts.

In the next section we describe the procedure of grouping our low-level features to built a mid-level representation of the video. We then use this to propose our mid-level part-based representation, which we then use as a basis for action recognition and localization.
7.2 Mid-level Representation

Given a sequence of images, we construct our basic low-level description as described in Chapter 6. Our low-level “tracklet descriptor” is designed to be insensitive to coarse variability in illumination, pose and location and retains as much as possible of the temporal evolution of the data. Our trajectory-based descriptor yields a time series of histograms of gradient orientations (HoG) [Low99] and histograms of optical flow (HoF) [LP07] for the entire trajectory of a track. These are grouped and described collectively as we explain in the next two sections.

7.2.1 Grouping trajectories

In order to aggregate trajectories of tracklet descriptors, we need to define a notion of similarity. Tracklets have a spatial motion component (the actual trajectory \(\{x[t]\}_{t=\tau}^T\)), a photometric component (HoG), and a shape (deformation) component (HoF), each with variable end-points \(\tau, T\). The trajectory of a point that does not move has constant spatial motion component, the trajectory of a point whose motion is rigid, planar and fronto-parallel has a constant shape component, and one that is Lambertian and moving in static ambient illumination has a constant photometric component. Each component is potentially relevant for grouping; however, we focus only on the spatial motion component. A slight complication is due to the variable end-points. We employ the distance recently proposed by [BM10] to measure similarity of trajectories that co-exist in a time interval, are spatial neighbors and move with similar motion. The distance penalizes trajectories that have different velocity profiles even in a small subset of their temporal overlap. Given two trajectories \(\{x_a[t]\}_{t=\tau_a}^{T_a}\), \(\{x_b[t]\}_{t=\tau_b}^{T_b}\) that co-exist in \([\tau_1, \tau_2]\), we have

\[
d(a, b) = \frac{1}{\tau_2 - \tau_1} \sum_{t=\tau_1}^{\tau_2} d_{\text{spatial}}[t] \cdot \max_{t \in [\tau_1, \tau_2]} d_{\text{velocity}}[t]
\]  

(7.1)
where \( d_{\text{spatial}}[t] = |x_a[t] - x_b[t]|_2 \) is the Euclidean distance of the trajectories at corresponding times and \( d_{\text{velocity}}[t] = |\dot{x}_a[t] - \dot{x}_b[t]|_2 \) is the \( \ell_2 \) distance of the velocity estimates, obtained by differentiation: \( \dot{x}[t] = x[t] - x[t-1] \). This is slightly different to [BM10] in that we do not normalize the distance between velocities with the local flow variation, since we use sparse feature tracks instead of a dense flow field.

To group trajectories we compute an affinity \( w(a,b) = \exp(-d(a,b)) \) between each pair \( a,b \), and form an \( n \times n \) affinity matrix for a video containing \( n \) trajectories. We then cluster trajectories using Affinity Propagation [FD07], that returns a membership indicator function \( m(\cdot) \). The only parameter of this algorithm is the ‘preference’ \( p \) for each node, which determines the number of clusters \( N \). To determine the appropriate number of clusters in a video sequence, we used Cattell’s scree test. Specifically, we set the number of clusters to \( N = \arg\min_i \lambda_i^2 + 1/\left( \sum_{j=1}^i \lambda_j^2 \right) + c \cdot i \), where \( \lambda_i \) are the eigenvalues of the affinity matrix, and \( c = 10^{-4} \). Bisection search over \( p \) was used to select the appropriate values of \( p \). This produces bundles of trajectories, illustrated in Fig. 7.1.

### 7.2.2 Describing trajectory groups

To describe a spatio-temporal volume covered by a trajectory group, rather than using holistic descriptors [WNL09, MHS10], or a simpler transition operator [YPY10], we wish to retain the locality of the descriptor of each individual trajectory in the group. Therefore, we “soft-assign” each tracklet descriptor to its 4-nearest neighbor codewords proportionally to their similarities, using a dictionary element learned, again, using Affinity Propagation, as done in Chapter 6. This yields a histogram \( h_k \) of tracklet codeword occurrences over the members of the group. Additionally, we apply Laplace smoothing to the histograms to prevent undesirable zero codeword probabilities. Each cluster thus retains information on the different component trajectories, that in turn
Figure 7.1: *Examples of trajectory groups: each group has a distinct color.*

capture the statistics of the spatio-temporal domain of the image in its moving frame.

To capture the coarser spatio-temporal shape characteristics of the ensemble of trajectories $\mathbf{x}_i$ in the group $k$ defined by their absolute positions: $D_k = \{\{\mathbf{x}_i[t]\}_{t=\tau_i}, \forall i : m(i) = k\}$, we compute the mean group trajectory:

$$g_k[t] = \frac{1}{|\{i\}|} \sum_{\forall i : t \in [\tau_i, T_i], \quad m(i) = k} \mathbf{x}_i[t] , \quad t \in \bigcup_{\{i : m(i) = k\}} [\tau_i, T_i]$$  \hspace{1cm} (7.2)

The mean group trajectories are used to estimate the pairwise relationships between groups within a video sequence, as described in Section 7.3.1. Collecting these descriptors, we have that each group $G_k$ is described by a pair $G_k = \{h_k, g_k\}$.
Moreover, at the coarsest level we form a simple bag-of-features (BoF) representation of all groups in terms of a histogram $h_0$ of all the tracklet descriptors extracted from the video. Consequently, a video $S$ can be described as the collection of the groups in combination with the histogram $h_0$: $S = \{h_0, \{G_k\}_{k=1}^N\}$.

7.3 Mid-level part model

The descriptor $S$ described above constitutes the basis of our mid-level action model. Our model is inspired from the constellation model [WWP00], the hidden Conditional Random Field models of [WM10] and the latent Support Vector Machine (SVM) model of [FGM10]. Apart from introducing a method that relies on a mid-level, sparse representation of a video we also introduce novel ideas with respect to these works at the modeling level, including among others the form of the pairwise features, the treatment of scale, and the Large-Margin ranking objective training procedure [Joa02, BVZ10], as detailed in Section 7.3.1. We detail how inference is used for classification in Section 7.3.2, describe model training in Section 7.3.3 and experimentally validate our model on activity recognition in Section 7.4.

7.3.1 Modeling group interactions

To capture the spatial context of an action, event, or activity, we leverage on the relation among mid-level parts using a graphical model. We use a fully connected graph $G = (V, E)$, with each node $i \in V$ encoding a part and each edge $(i, j) \in E$ encoding pairwise relations between parts. An additional isolated node $F$ represents the video as a whole.

Given a video $x$ that has been decomposed into $N$ clusters we consider a vector of discrete latent variables $P = [p_1, \ldots, p_{|V|}]$, with $p_i \in \{1, \ldots, N\}$ that associates
each node $i$ with one of the $N$ clusters. We employ one more latent variable $\sigma$, shared among all spatio-temporal relations; this allows us to bring the relative locations of the action parts to a canonical scale and deal with the nuisance parameter of the action’s scale in a video.

The latent variable vector thus identifies the locations and appearances of the action parts in the video; each action part is ‘instantiated’ with the location and appearance of its assigned trajectory group. Conditioned on the latent variable vector we can score a video using our model’s unary and pairwise terms, which capture appearance and spatio-temporal information respectively.

The $i$-th unary term $u_i$ scores the tracklet descriptors $h_{pi}$ computed for cluster $p_i$ using a linear kernel as $u_i = \langle w_i, h_{pi} \rangle$. The parameters $w_i$ are estimated discriminatively and allow each part to be tuned to different mid-level action properties. The isolated node $F$ has a score $u_F = \langle w_0, h_0 \rangle$ where $h_0$ is the bag-of-words histogram.

The pairwise term $u_{i,j}$ captures the spatio-temporal relation of the action parts $i, j$. If nodes $i$ and $j$ have been matched to groups $G_{pi}, G_{pj}$ with mean group trajectories $g_{pi} \in \mathbb{R}^{2 \times \bar{T}_{pi}}, g_{pj} \in \mathbb{R}^{2 \times \bar{T}_{pj}}$, we first estimate if the mean group trajectories co-exist for a sufficiently long time interval. If this condition is not satisfied their pairwise feature is set to $\psi(g_{pi}, g_{pj}, \sigma) = 0$. Otherwise a feature describing the evolution of their relative positions is computed. First, the canonical relative position is estimated $d_{pi,pj,\sigma}[\bar{t}] = (g_{pi}[\bar{t}] - g_{pj}[\bar{t}]) / \sigma$, where $\sigma$ is the latent scale variable, and $\bar{t}$ takes values only in the coexisting time interval. Subsequently, the velocity of change of the absolute relative position is calculated $v^{x,y}[\bar{t}] = |d_{pi,pj,\sigma}[\bar{t} + 1]| - |d_{pi,pj,\sigma}[\bar{t}]|$. Note that this measure is independent of ordering of $p_i$ and $p_j$. We then soft-quantize the individual coordinates
x, y of \( v^{x,y}[\tilde{t}] \) using a \( 2n + 3 \)-dimensional vector \(^1\):

\[
\overline{\phi}(x) = [v_{-1}(x), \rho_{-n}(x), \ldots, \rho_0(x), \ldots, \rho_n(x), v_1(x)], \quad v_{\pm 1}(x) = \left(1 + e^{\frac{x \pm \mu_0}{s_0}}\right)^{-1}
\]

This ‘soft binning’ vector is extracted separately for each coexisting time instance \( \tilde{t} \) which gives us two matrices \( \overline{v}_x, \overline{v}_y \in \mathbb{R}^{(2n+3) \times \overline{T}_{p_i,p_j}} \) for each coordinate, where \( \overline{T}_{p_i,p_j} \) is the length of the coexisting interval. Finally, we set our pairwise feature \( \psi(g_{p_i}, g_{p_j}, \sigma) \) equal to the vectorized result of \( \overline{v}_x \overline{v}_y^T \). This feature vector can indicate, for instance, whether the two groups are converging in the \( x \) coordinate and diverging in the \( y \) coordinate. The pairwise potential is obtained as the inner product with a weight vector \( w_{i,j} \), \( u_{i,j} = \langle w_{i,j}, \psi(g_{p_i}, g_{p_j}, \sigma) \rangle \).

Summarizing, once the latent variables \( z = \{\sigma, P\} \) are given, we can quantify the fit of a video to our action model in terms of a cost obtained by adding the corresponding unary and pairwise terms:

\[
\text{score}(z) = \langle w_0, h_0 \rangle + \sum_{i=1}^{\left|V\right|} \langle w_i, h_{p_i} \rangle + \sum_{i=1}^{\left|V\right|} \sum_{j=i+1}^{\left|V\right|} \langle w_{i,j}, \psi(g_{p_i}, g_{p_j}, \sigma) \rangle,
\]

Having formulated our model, we now turn to the two main problems of (i) estimating the optimal \( z \), given a video and the model parameters and (ii) estimating the model parameters from training data.

### 7.3.2 Classification by Subgraph Matching

Classifying a video based on the score described in Equation 7.4 entails maximizing it over \( z = (P, \sigma) \) i.e., our discriminant function is \( s = \text{argmax}_z \text{score}(z) \). As the num-

\(^1\)The parameters in this feature vector are fixed for all actions. The \( \mu, s \) parameters are such that the Gaussian functions \( \rho_\eta \) tessellate the velocity axes geometrically, \( \mu_\eta = r^\eta \mu_0, s_\eta = \alpha \mu_0 \), while \( s_0 \) and \( \mu_0 \) are set so that the sigmoidal functions \( v \) cover the extremes of the domain.
ber of clusters is larger than the number of parts, finding the best cluster-part assignment amounts to solving subgraph matching, a combinatorial optimization problem that emerges in a range of applications.

If the cost function contained only unary terms, estimating $P$ for known $\sigma$ would amount to solving a linear assignment problem, which can easily be solved using Linear Programming. As our cost function includes pairwise terms, we face the NP-hard Quadratic Integer Programming problem. Approximate solutions for this problem in vision include LP/SDP relaxations [SS03], spectral methods [LH05, CSS07], MRF inference [TKR08, KP08]. We follow this last thread and use the TRW-S method of [Kol06] which was shown to perform marginally worse than the state-of-the-art method of [TKR08] in substantially less time - for our problem it takes a fraction of a second.

To formulate subgraph matching as an MRF labeling problem for each node $i$ we consider the unary cost $u_i(p_i)$ incurred if we assign to it a label $p_i \in \{1, \ldots, N\}$ and the pairwise cost $u_{i,j}(p_i, p_j)$ paid for each of its neighbors $j$ and their possible nodes $j$. Using the following expressions for the unary and pairwise terms:

$$u_i(p_i) = \langle w_i, h_{p_i} \rangle,$$

$$u_{i,j}^\sigma(p_i, p_j) = \begin{cases} 
\langle w_{i,j}, \psi(g_{p_i}, g_{p_j}, \sigma) \rangle, & p_i \neq p_j \land \psi \neq 0 \\
-\infty, & p_i = p_j \lor \psi = 0 
\end{cases}$$

Equation 7.4 writes $\text{score}(z) = \sum_{i \in V} u_i(p_i) + \sum_{(i,j) \in E} u_{i,j}^\sigma(p_i, p_j)$, a standard MRF energy, apart from the scale variable. To optimize over $z = (\sigma, P)$ we consider a discrete set of scale values, $\sigma \in \{\sigma_1, \ldots, \sigma_{N'}\}$; for each $\sigma_k$ we estimate $P_k^* = \text{argmax} \text{score}(P, \sigma_k)$ using TRW-S. We finally choose the scale index $k^* = \text{argmax}_k \text{score}(P_k^*, \sigma_k)$ that yields the smallest energy. The output of this process is the latent variable vector $z^* = (P_k^*, \sigma_{k^*})$ that best fits the action model to a video.
7.3.3 Learning

We now address the issue of learning the parameters of our score function. Our score for a video \( x_i \) is the inner product \( \langle w, \phi(x_i, z_i) \rangle \) of the feature \( \phi(x_i, z_i) \) and the parameter vector \( w \), where

\[
\phi(x_i, z_i) = [h_0, h_p1, \ldots, h_{p|V|}, \psi(g_{p1}, g_{p2}, \sigma), \ldots, \psi(g_{p|V|}, g_{p|V|-1}, \sigma)],
\]

\[
w = [w_0, w_1, \ldots, w_{|V|}, w_{|V|,2}, \ldots, w_{|V|,|V|-1}].
\]

In the previous section, we described how to optimize the score(\( z; w \)) over \( z \) for known \( w \). Our task now is to find the \( w \) that leads to the maximum margin classification. This is equivalent to minimizing the sum of a convex and a concave function, whose optimal solution can be approximated using an alternating optimization algorithm such as CCCP [YR03], which we adopt. Specifically, the learning procedure alternates between maximizing the score function over the latent variables for each positive and negative sample, and minimizing the SVM objective over the parameter vector \( w \).

However, when we employed this scheme, we noticed that the SVM objective was affected by the imbalance between the number of positive and negative examples. Consequently, the algorithm focused on satisfying the constraints of the negative samples, neglecting the constraints on the positive samples. The same empirical observation has also been reported in [BVZ10]. Hence, as in [BVZ10], we address this issue by adopting the ranking SVM algorithm [Joa02] in lieu of the traditional SVM objective:

\[
\begin{align*}
\min_{w, \xi} & \quad \frac{1}{2} \|w\|^2 + C \sum_{i,j} \xi_{ij} \\
\text{s.t.} & \quad \langle w, \phi(x_i, z_i^*) \rangle - \langle w, \phi(x_j, z_j^*) \rangle \geq \Delta_{0/1}(y_i, y_j) - \xi_{ij}, \forall i, j, y_j \in \mathcal{Y}\{y_i\} \\
& \quad \xi_{ij} \geq 0 \forall i, j
\end{align*}
\]

(7.5)

For initialization, we set the pairwise weights \( w_{i,j} \) to zero and use the following initialization of the unary term weights \( w_i \): each \( w_i \) is set equal to the center of a cluster.
produced by K-means on the collection of vectors \( h_k \) of all the positive training videos.

### 7.4 Experiments

We validate our model on the Hollywood1 Human Action benchmark dataset (HOHA) [LMS08]. This dataset contains 430 videos (240 × 450, 24 fps). The HOHA dataset is extremely challenging as each video sequence, in addition to the action being performed, contains nuisances such as significant camera motion, rapid scene changes and occasionally significant clutter. Moreover, the actions it includes (e.g. “sit down”, or “kiss”) can manifest themselves in a wide variety of conditions, only a tiny portion of which are sampled in the training set, making the classification task extremely challenging. Furthermore, many actions are not characterized by a single agent (such as “sit down”) but involve interactions with other agents (“kiss”) or objects (“get out of car”). We chose to evaluate our method on HOHA instead of other benchmark datasets because for some of them, e.g. KTH [SLC04], Weizmann [BGS05], one can already get extremely high performance even with simple bag-of-features techniques (Chapter 6), and others, e.g. UCF Sport, allow high-performance classification using background cues [WUK09].

In order to be able to: a) quantify our localization performance, b) aid the training phase of the model, by restricting the possible selections of parts to trajectory groups relevant to the action, we annotated the HOHA dataset with bounding boxes at each frame around the action of interest (Fig. 7.3). This weak supervision improves our algorithm’s ability to learn meaningful parts, enhancing our recognition performance. This can be attributed to the large variability among action instances and the limited size of the training set. We emphasize that this weak annotation is not used in the testing phase of the algorithm.
**Experimental settings:** for all classes, we use $|V| = 4$ graph nodes. For the pairwise relations described in Equation 7.3.1, we use $n = 3$ while $\mu, s_\eta$ are set to cover the relative velocity domain spanned by the database videos. The penalty parameter $C$ of the ranking objective is selected with 5-fold cross-validation in the training set. We consider 6 values for the latent variable $\sigma$, logarithmically spaced in the interval $[0.5, 2]$. The histograms $h_o$ and $h_k$ of tracklet codeword occurrences are mapped via the approximate feature map for the $\chi^2$ kernel [VZ10]. This allows us to combine the increased discriminative ability of the $\chi^2$ with the efficient training and testing of linear kernels.

**Action Recognition.** We evaluate our model following the experimental setting proposed by [LMS08]: The test set has 211 videos with 217 labels and the training set has 219 videos with 231 labels (manually annotated). For each action, we train our model and evaluate our performance with average precision (AP) on the precision/recall curve. Table 7.1 shows the performance of our model using either the unary terms of each *part* alone or both pairwise and unary terms. Moreover, as mentioned above, we observe a boost in the performance when we discarded the candidate trajectory groups in the training set that had no overlap with the bounding boxes. We call this approach the restricted training approach (Table 7.1, first two columns). Another observation is that if we use the regular SVM objective the mean AP of our method is 35.5%, as oppose to 38.0% using the ranking SVM. Table 7.2 summarizes the results of our full model along with competing approaches for comparison. Our scheme is competitive with most schemes and performs best in three categories (sit down, stand up, hand shake), comparable to the best in other categories, and worse than Laptev *et al.* [LMS08] on two categories, which gives [LMS08] a marginal edge over our approach. The lower performance in those categories can be attributed to the fact that in many of their videos the amount of extracted trajectories are fewer when compared to the better performing categories. This leads to fewer trajectory groups and very sparse
feature histograms, which affect the classifier. However, it should be noted that the multi-grid spatial binning approach of [LMS08] cannot be used in support of other tasks, such as the localization that our model performs. An outlier in the comparison is the approach of Sun et al. [SWY09], whose results we could match only with a global approach manually tuned for each category. We believe that our approach is more flexible and transparent, allowing analysis of each component, yielding insights into its role in the overall performance.

**Action Localization.** To evaluate the relevance of the selected trajectory groups to the performed action, we define the localization score:

\[
\frac{1}{|V|} \sum_{i=1}^{|V|} \sum_{t=1}^{T} \frac{|D_{i,t} \cap L_t|}{|D_{i,t}|} \geq \theta,
\]

where \(\cdot\) is the zero-one indicator function, \(L_t\) the set of points inside the annotated bounding box, \(D_{i,t}\) the set of points belonging to the selected trajectory group and \(\theta\) is the threshold that defines the minimum overlap of trajectories of a group to consider it as a part of the bounding box. Fig. 7.2 illustrates the average localization score across the test videos of each action, as well as the mean localization score across all actions.

From this figure we notice that even for the maximum threshold \(\theta = 1\) (i.e., all the points of the trajectory group lie inside the bounding box at a given time instance), we get a localization score of 29.56%. This shows that our method is able to select meaningful trajectory groups as parts (Fig. 7.3). To the best of our knowledge, no such localization results have been reported before for this challenging dataset.
Table 7.1: Performance comparison on HOHA Dataset, using different components of our model.

<table>
<thead>
<tr>
<th>Class</th>
<th>Pairwise &amp; Unary Terms - Restricted Training</th>
<th>Only Unary Terms - Restricted Training</th>
<th>Pairwise &amp; Unary Terms - Without Restriction</th>
<th>Only Unary Terms - Without Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answer phone</td>
<td>24.5%</td>
<td>28.5%</td>
<td>25.0%</td>
<td>28.8%</td>
</tr>
<tr>
<td>Get out of car</td>
<td>39.3%</td>
<td>36.8%</td>
<td>35.8%</td>
<td>35.7%</td>
</tr>
<tr>
<td>Hand shake</td>
<td>35.0%</td>
<td>32.5%</td>
<td>31.5%</td>
<td>32.4%</td>
</tr>
<tr>
<td>Hug person</td>
<td>25.1%</td>
<td>25.0%</td>
<td>24.3%</td>
<td>24.5%</td>
</tr>
<tr>
<td>Kiss</td>
<td>56.8%</td>
<td>56.6%</td>
<td>55.7%</td>
<td>56.5%</td>
</tr>
<tr>
<td>Sit down</td>
<td>45.9%</td>
<td>35.2%</td>
<td>38.0%</td>
<td>32.9%</td>
</tr>
<tr>
<td>Sit up</td>
<td>17.5%</td>
<td>17.3%</td>
<td>17.3%</td>
<td>17.5%</td>
</tr>
<tr>
<td>Stand up</td>
<td>60.0%</td>
<td>59.4%</td>
<td>58.8%</td>
<td>60.4%</td>
</tr>
<tr>
<td>Mean AP</td>
<td>38.0%</td>
<td>36.4%</td>
<td>35.8%</td>
<td>36.1%</td>
</tr>
</tbody>
</table>

Table 7.2: Performance comparison on HOHA Dataset.

<table>
<thead>
<tr>
<th>Class</th>
<th>Our Model</th>
<th>Tracklet descriptor</th>
<th>Laptev et al. [LMS08]</th>
<th>Yeffet et al. [YW09]</th>
<th>Matikainen et al.</th>
<th>Kläser et al. [KMS08]</th>
<th>TTD Combined</th>
<th>TTD-SIFT Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answer phone</td>
<td>24.5%</td>
<td>26.7%</td>
<td>26.7%</td>
<td>32.1%</td>
<td>35.1%</td>
<td>35.0%</td>
<td>18.6%</td>
<td></td>
</tr>
<tr>
<td>Get out of car</td>
<td>39.3%</td>
<td>28.1%</td>
<td>22.5%</td>
<td>41.5%</td>
<td>32.6%</td>
<td>7.7%</td>
<td>22.6%</td>
<td></td>
</tr>
<tr>
<td>Hand shake</td>
<td>35.0%</td>
<td>18.9%</td>
<td>23.7%</td>
<td>32.3%</td>
<td>33.8%</td>
<td>5.3%</td>
<td>11.8%</td>
<td></td>
</tr>
<tr>
<td>Hug person</td>
<td>25.1%</td>
<td>25.0%</td>
<td>34.9%</td>
<td>40.6%</td>
<td>28.3%</td>
<td>23.5%</td>
<td>19.8%</td>
<td>N/A</td>
</tr>
<tr>
<td>Kiss</td>
<td>56.8%</td>
<td>51.5%</td>
<td>52.6%</td>
<td>53.3%</td>
<td>57.6%</td>
<td>42.9%</td>
<td>47.0%</td>
<td></td>
</tr>
<tr>
<td>Sit down</td>
<td>45.9%</td>
<td>23.8%</td>
<td>37.8%</td>
<td>38.6%</td>
<td>36.2%</td>
<td>13.6%</td>
<td>32.5%</td>
<td></td>
</tr>
<tr>
<td>Sit up</td>
<td>17.5%</td>
<td>23.9%</td>
<td>15.2%</td>
<td>18.2%</td>
<td>13.1%</td>
<td>11.1%</td>
<td>7.0%</td>
<td></td>
</tr>
<tr>
<td>Stand up</td>
<td>60.0%</td>
<td>50.1%</td>
<td>45.4%</td>
<td>50.5%</td>
<td>58.3%</td>
<td>42.9%</td>
<td>38.0%</td>
<td></td>
</tr>
<tr>
<td>MAP</td>
<td>38.6%</td>
<td>32.1%</td>
<td>32.9%</td>
<td>38.4%</td>
<td>36.8%</td>
<td>22.8%</td>
<td>24.7%</td>
<td>30.3%</td>
</tr>
</tbody>
</table>
Figure 7.2: Localization scores for the trajectory groups selected by our algorithm as function of the overlap threshold ($\theta$). Notice that the mean of localization score across all actions is 29.56% for a high threshold ($\theta = 1$) and 59.12% for a low threshold ($\theta = 0.1$).

7.5 Discussion

In this chapter, we have presented an approach to modeling spatio-temporal statistics of video for the purpose of classification of actions, events or activities. Starting from local spatio-temporal descriptors from the literature, we assemble a mid-level model of individual spatio-temporal tracks and their pairwise relations. Our model lends itself
for use in standard classification schemes; specifically, we use a latent SVM framework to simultaneously learn the parameters of our models and perform classification.

Testing such models is not straightforward. We use the HOHA benchmark which poses several challenges, both due to its limited number of training examples and due to the diversity of the actions. While our average performance is below the current best multi-grid global scheme, it is comparable or better than the rest of the existing approaches, and in particular better than all local models. Moreover, part-based models are desirable in action recognition because they support a variety of other tasks beyond straight classification of an entire video shot into one of a few action categories. For instance, we show that they enable localization, by flagging the local components (parts) that are most discriminative.

Since our model relies on extracted descriptors, its performance degrades when the low-level data are uninformative, for instance in shots that are too short or too dark and thus yield no discriminative low-level features. Moreover, the huge variability of real-world actions can be barely captured by the small number of instances appearing in current benchmark datasets. We believe that, as larger and richer datasets become available, featuring a sufficient number of training data for each action or action component, the localization power of our approach will pay off, especially when used to localize not entire complex actions such as “kiss” but simpler action-segments (‘turn key’) from which more complex actions (‘open door’) can be composed.
Figure 7.3: Sample frames from different video sequences of the test set across different actions. The colored trajectories represent selected trajectory groups identified by our algorithm. The color indicates the node association in our model. The white trajectories illustrate the trajectories that were not selected as parts. From these figures, we can observe that the selected trajectory groups lie within the manually annotated bounding boxes, shown in cyan.
CHAPTER 8

Summary and Discussion

In this thesis, we have explored methods for recognizing human actions using different modalities, such as video, motion capture data and depth cameras. We have focused on developing algorithms that model explicitly through dynamical models or implicitly through features the temporal evolution of different human activities. In this chapter we summarize our findings.

Feature extraction in time series. In Chapter 3, we proposed a method for extracting salient features from time series data, such as motion capture data and extracted human silhouettes from video sequences. We have found that complex human actions can be decomposed into simple primitive actions. Specifically, the underlying motion can be represented by a low dimensional time series. We have seen that constructing the features dictionary that exploits the temporal multi-scale characteristics of the motion primitive leads to a more discriminative representation of action. These multi-scale features in combination with kernel based classification techniques that account for the temporal ordering of the extracted features outperform existing action recognition methods on standard benchmark datasets.

Input driven dynamical models. In Chapter 4, we have developed algorithms that model motion capture data as the output of linear dynamical systems (LDSs). As opposed to existing techniques that use noise driven LDSs or switched LDSs, our model uses a bounded sparse one dimensional input signal to drive the LDS. Since this input is unknown, we developed an algorithm for inferring both the unknown
input as well as the model parameters of the LDS, using an alternating optimization scheme. We have found that inferred input signal contains different “spike” patterns for different actions enabling unsupervised segmentation. Although, the “spike” patterns are discriminative enough to perform unsupervised segmentation of human motion, they are not as rich representation as the one proposed in Chapter 3. Consequently, the recognition accuracy in the supervised setting is a bit lower than the method proposed in Chapter 3.

**Real-time action classification using depth sensor.** In Chapter 5, we proposed a classification scheme which is based on the angular representation of the skeleton extracted from real time depth camera [SFC11]. This classifier is a cascaded correlation-based classifier and it can be interpreted as a template matching scheme, where a combination of a naive Bayes classifier followed by a logistic regression classifier account for the variability of the measured action and its corresponding exemplar. In order to succeed real-time performance we make a strong assumption, but in several realistic scenarios valid, that the input motion adheres to a known, canonical time-base: a musical beat. This enables the correlation of the template not to be performed in a sliding window approach but in well defined time segments. We tested the algorithm succeed excellent classification performance on a dataset comprising of thousands action instances recorded using XBOX Kinect platform.

**Trajectory based features.** In Chapter 6, we proposed a spatio-temporal feature descriptor that captures the local structure of the image around trajectories tracked over time. Similar to the features proposed in Chapter 3, the proposed feature descriptor encodes implicitly the temporal evolution of the underlying motion. Moreover, we advocate that extracting spatio-temporal feature descriptors of varying temporal length based on the “trackability” of the local region enhances the discriminative power of the video representation. Based on our experimental validation, we have found that our
feature which describes the evolution of spatial structures in time is more informative than existing spatio-temporal descriptors that describe a cube around an interest point. This can be attributed to the fact that when one describes a spatio-temporal cube, it could contain multiple structures.

**Mid-level part based representation.** In Chapter 7, we have developed a mid-level representation of human actions that is based on grouping the low level features that we introduced in Chapter 6. Based on this grouping, the video is decomposed in a set of spatio-temporal segments that are potential candidates of being a “part” relevant to the performed action. We have found that the proposed action model is able to neglect the majority of the regions in a video sequence and just encode the characteristics of each individual part and additionally their pairwise spatio-temporal structure. As a consequence, when compared to the simple bag-of-features classification strategy, our approach shows a better performance at recognizing actions. Thus capturing both the temporal of the parts as well as the global configuration between them is important for action recognition.

We have made publicly available the implementations of our work (Chapter 4, Chapter 6). In addition, code is also available for optical flow with occlusion estimation algorithm, which we have outline in detail in the appendix. This contribution is fundamental for the problem of optical flow estimation, which is one of the more important low level cues that we have used as a part of our trajectory descriptor (Chapter 6).
APPENDIX A

Sparse Occlusion Detection with Optical Flow

Occlusion phenomena are a critical component of the image formation process, shaping the statistics of natural images. Occlusion detection plays an important role in priming visual recognition of detached objects [AS11], like a person moving, in navigation and interaction with natural environments, and more in general in the visual information-gathering process: The complexity of the image in the “discovered region” is related to the *Actionable Information Increment* in visual exploration [Soa11].

Occlusions arise when a portion of the scene is visible in one image, but not another. In a video-stream, occlusions typically occur at depth discontinuities. We are interested in determining the *occluded regions*, that is the subset of an image domain that back-projects onto portions of the scene that are not *co-visible* from a temporally adjacent image.\(^1\) The occluded region is, in general, multiply connected, and can be quite complex, as the example of a barren tree illustrates.

Portions of the scene that are *co-visible* can be mapped onto one another by a domain deformation [SPV09], called *optical flow*. It is, in general, different from the *motion field*, that is the projection onto the image plane of the spatial velocity of the scene [VP89], unless three conditions are satisfied: (a) Lambertian reflection, (b) constant illumination, and (c) constant visibility properties of the scene. Most surfaces with benign reflectance properties (diffuse/specular) can be approximated as Lambertian

\(^1\)This process could be generalized to global co-visibility, resulting in a model of the world with topologically distinct “layers” [WA94]. This is beyond the scope of this appendix and has already been addressed in a variational setting, the first example being [JYS05, JYS08].
almost everywhere under sparse illuminants (e.g., the sun). In any case, widespread violation of Lambertian reflection does not enable correspondence, so most optical flow methods embrace (a), either implicitly or explicitly. Similarly, constant illumination (b) is a reasonable assumption for ego-motion (the scene is not moving relative to the light source), and even for objects moving (slowly) relative to the light source. Assumption (c) is needed in order to have a dense flow field: If an image contains portions of its domain that are not visible in another image, these can patently not be mapped onto it by optical flow vectors; (c) is often assumed because optical flow is defined in the limit where two images are sampled infinitesimally close in time, in which case there are no occluded regions, and one can focus solely on discontinuities\(^2\) of the motion field. Thus, the great majority of variational motion estimation approaches provide an estimate of a dense flow field, defined at each location on the image domain, including occluded regions. In their defense, it can be argued that for small parallax (slow-enough motion, or far-enough objects, or fast-enough temporal sampling) occluded areas are small. However, small does not mean absent, nor unimportant, as occlusions are critical to perception [Gib84] and a key for developing representations for recognition. For this reason, we focus on occlusion detection in video streams.

Occlusion detection would be easy if the motion field was known. Vice-versa, optical flow estimation would be easier if the occluded domain was known. As often in vision problems, one knows neither, so in the process of inferring the object of inference (the occluded domain) we will estimate optical flow, the “nuisance variable,” as a byproduct.

In this appendix we (I) show that, starting from the standard assumptions (a)-(b), the problem of detecting (multiply-connected) occlusion regions can be formulated as

\(^2\)In occluded regions, the problem is not that optical flow is discontinuous; it is simply not defined; it does not exist. Motion in occluded regions can be hallucinated or extrapolated, based on the prior or regularizer. However, whatever motion is assigned to an occluded region cannot be validated from the data.
a variational optimization problem (Section A.1). We then (II) show how the functional to be minimized can be relaxed into a sequence of convex functionals and minimized using \textit{re-weighted} \( \ell_1 \) optimization (Equation A.16). At each iteration, the functional to be minimized is related to those used for optical flow estimation, but the minimization is with respect to the indicator function of the occluded region, not just the (dense) optical flow field. We then bring to bear two different approaches to optimize these functionals, one is (III) an optimal first-order method, due to Nesterov (Section A.2), and the other one is (IV) an alternating minimization technique, known as split-Bregman method (Section A.3). We evaluate our approach empirically in Sections A.4 and A.0.2, and discuss its strengths and limitations of in Section A.5.

To the best of our knowledge, neither the formulation of occlusion detection and motion estimation as a joint minimization problem under sparsity prior on the occluded regions (I), nor the use of re-weighted \( \ell_1 \) (II), Nesterov’s algorithm (III), or split-Bregman method (IV) have ever been presented before in the optical flow literature. A preliminary version of the algorithms presented here has appeared in [ARS10].

A.0.1 Prior Related Work

Several algorithms have been proposed to perform occlusion detection. Many define occlusions as the regions where forward- and backward-motion are inconsistent [PVO94, ADP07]. This is problematic as the motion in the occluded region is not just inconsistent, it is \textit{undefined}, as there is no “motion” (domain deformation) that takes one image onto another. Other approaches [LDC02, KZ01, SLK05] formulate occlusion directly as a classification problem, and perform motion estimation in a discrete setting, where it is an NP-hard problem. This can then be approximated with combinatorial optimization. Others [IK08, BS07] also exploit motion symmetry to detect occlusions and weight the inconsistencies with a monotonically decreasing function.
The residual from optical flow estimation has also been used to decide whether a region is occluded. Strecha et al. [SFV04] proposed a probabilistic formulation to detect occluded regions using the estimated noise model and histogram of occluded pixel intensities. Xiao et al. [XCS06] threshold the residual obtained using level-set methods to find occluded areas. Both try to minimize a non-convex energy function by iterating between two subproblems, occlusion detection and motion estimation.

Another set of algorithms infer occlusion boundaries [SH09, HY10] and occluded regions [HMB11] training a learning based detector using appearance, motion and depth features. The accuracy of these methods largely depends on the performance of underlying feature detectors. Furthermore, these methods require that the optical flow is estimated beforehand. Such a divide-et-impera approach comes at the cost of overall optimality where as we focus on a method that jointly estimates flow fields and detects occluded areas.

Occlusions have also been a concern in the optical flow community since the first global formulation was proposed by Horn and Schunck [HS81]. Black and Anandan [BA96] proposed replacing the $\ell_2$ norm of the residual with a non-convex Lorentzian penalty. Another common criterion used for this purpose is the $\ell_1$ norm of the residual, which is non-trivial to minimize since it is non-smooth. [BWS05, BBP04] use Charbonnier’s penalty that is a differentiable approximation of the $\ell_1$ norm; others [WPZ08, WCP09, WTP09] solved the non-smooth problem with primal-dual methods decoupling the matching and regularization terms. However, none of these robust flow estimation methods focus on the detection of occlusions.

A.0.2 Evaluation

Optical flow estimation is a mature area of computer vision, and benchmark datasets have been developed, the best known example being the Middlebury [BSL07]. Un-
fortunately, no existing benchmark provides ground truth for occluded regions, nor a scoring mechanism to evaluate the performance of occlusion detection algorithms. Unfortunately, this also biases the motion estimation scoring mechanism as ground truth motion is provided on the entire image domain, including occluded regions, where it can be extrapolated using the priors/regularizers, but not validated from the data.

To overcome this gap, we have produced a new benchmark by taking a subset of the training data in the Middlebury dataset, and hand-labeling occluded regions. We then use the same evaluation method of the Middlebury for the (ground truth) regions that are co-visible in at least two images. This provides a motion estimation score. Then, we provide a separate score for occlusion detection, in terms of precision-recall curves. This dataset (that at the moment is limited by our ability to annotate occluded regions to a subset of the full Middlebury, but that we will continue to expand over time), as well as the implementation of our algorithm in source format will be released publicly after the anonymous review process is successfully completed.

\section*{A.1 Joint Occlusion Detection and Optical Flow Estimation}

In this section, we show how the assumptions (a)-(b) can be used to formulate occlusion detection and optical flow estimation as a joint optimization problem. We assemble a functional that penalizes the (unknown) optical flow residual in the (unknown) co-visible regions, as well as the area of the occluded region. The resulting optimization problem has to be solved jointly with respect to the unknown optical flow field, and the indicator function of the occluded region.

Let $I : D \subset \mathbb{R}^2 \times \mathbb{R}^+ \rightarrow \mathbb{R}^+; (x, t) \mapsto I(x, t)$ be a grayscale time-varying image defined on a domain $D$. Under the assumptions (a)-(b), the relation between
two consecutive frames in a video \( \{ I(x,t) \}_{t=0}^{T} \) is given by

\[
I(x,t) = \begin{cases} 
I(w(x,t), t + dt) + n(x,t), & x \in D \setminus \Omega(t; dt) \\
\rho(x,t), & x \in \Omega(t; dt) 
\end{cases}
\]

\[
(A.1)
\]

where \( w : D \times \mathbb{R}^+ \to \mathbb{R}^2; x \mapsto w(x,t) = x + v(x,t) \) is the domain deformation mapping \( I(x,t) \) onto \( I(x,t + dt) \) everywhere except at occluded regions. Usually \textit{optical flow} denotes the incremental displacement \( v(x,t) = w(x,t) - x \). The occluded region \( \Omega \) can change over time depending on the temporal sampling interval \( dt \) and is not necessarily simply-connected; so even if we call \( \Omega \) the occluded region (singular), it is understood that it can be made of several disconnected portions. Inside \( \Omega \), the image can take any value \( \rho : \Omega \times \mathbb{R}^+ \to \mathbb{R}^+ \) that is in general unrelated to \( I(w(x), t + dt) \). In the limit \( dt \to 0 \), \( \Omega(t; dt) = \emptyset \). Because of (almost-everywhere) continuity of the scene and its motion (i), and because the additive term \( n(x,t) \) compounds the effects of a large number of independent phenomena\(^3\) and therefore we can invoke the Law of Large Numbers (ii), in general we have that

\[
(i) \quad \lim_{dt \to 0} \Omega(t; dt) = \emptyset, \quad \text{and} \quad (ii) \quad n \overset{\text{IID}}{\sim} \mathcal{N}(0, \lambda)
\]

\[
(A.2)
\]

i.e., the additive uncertainty is normally distributed in space and time with an isotropic and small variance \( \lambda > 0 \). We define the residual \( e : D \to \mathbb{R} \) on the entire image domain \( x \in D \), via

\[
e(x,t; dt) = I(x,t) - I(w(x,t), t + dt)
\]

\[
(A.3)
\]

\[
e = \begin{cases} 
n(x,t), & x \in D \setminus \Omega \\
\rho(x,t) - I(w(x,t), t + dt), & x \in \Omega
\end{cases}
\]

\[\text{where } n(x,t) \text{ collects all unmodeled phenomena including deviations from Lambertian reflection, illumination changes, quantization error, sensor noise, and later also linearization error. It does not capture occlusions, since those are explicitly modeled.}\]
which we can write as the sum of two terms, $e_1 : D \to \mathbb{R}$ and $e_2 : D \to \mathbb{R}$, also defined on the entire domain $D$ in such a way that

$$
\begin{cases}
e_1(x, t; dt) = \rho(x, t) - I(w(x, t), t + dt), & x \in \Omega \\
e_2(x, t; dt) = n(x, t), & x \in D \setminus \Omega.
\end{cases}
$$

(A.4)

Note that $e_2$ is undefined in $\Omega$, and $e_1$ is undefined in $D \setminus \Omega$, in the sense that they can take any value there, including zero, which we will assume henceforth. We can then write, for any $x \in D$,

$$
I(x, t) = I(w(x, t), t + dt) + e_1(x, t; dt) + e_2(x, t; dt)
$$

(A.5)

and note that, because of (i) $e_1$ is large but sparse, while because of (ii) $e_2$ is small but dense. We will use this as an inference criterion for $w$, seeking to optimize a data fidelity term that minimizes the number of nonzero elements of $e_1$ (a proxy of the area of $\Omega$), and the negative log-likelihood of $n$.

$$
\psi_{\text{data}}(w, e_1) \doteq \|e_1\|_{L^0(D)} + \frac{1}{\lambda}\|e_2\|_{L^2(D)} \quad \text{subject to (A.5)}
$$

where $\|f\|_{L^0(D)} \doteq |\{x \in D | f(x) \neq 0\}|$ and $\|f\|_{L^2(D)} \doteq \int_D |f(x)|^2 dx$. Unfortunately, we do not know anything about $e_1$ other than the fact that it is sparse, and that what we are looking for is $\chi(\Omega) \propto e_1$, where $\chi : D \to \mathbb{R}^+$ is the characteristic function that is non-zero when $x \in \Omega$, i.e., where the occlusion residual is non-zero. So, the data fidelity term depends on $w$ but also on the characteristic function of the occlusion domain $\Omega$. For a sufficiently small $dt$, we can approximate, for any $x \in D \setminus \Omega$,

$$
I(x, t + dt) = I(x, t) + \nabla I(x, t)v(x, t) + n(x, t)
$$

(A.9)

\begin{footnotesize}
\footnote{Sparse stands for almost everywhere zero on $D$. Similarly, dense stands for almost everywhere non-zero.}
\end{footnotesize}

\begin{footnotesize}
\footnote{In a digital image, both domains $D$ and $\Omega$ are discretized into a lattice, and $dt$ is fixed. Therefore,}
\end{footnotesize}
where the linearization error has been incorporated into the uncertainty term \( n(x, t) \). Therefore, following the same previous steps, we have

\[
\psi_{\text{data}}(v, e_1) = \| \nabla I v + I_t - e_1 \|_{L^2(D)} + \lambda \| e_1 \|_{L^0(D)}. \tag{A.10}
\]

Since we typically do not know the variance \( \lambda \) of the process \( n \), we will treat it as a tuning parameter, and because \( \psi_{\text{data}} \) or \( \lambda \psi_{\text{data}} \) yield the same minimizer, we have attributed the multiplier \( \lambda \) to the second term. In addition to the data term, because the unknown \( v \) is infinite-dimensional and the problem is ill-posed, we need to impose regularization, for instance by requiring that the total variation (TV) be small

\[
\psi_{\text{reg}}(v) = \mu \| v_1 \|_{TV} + \mu \| v_2 \|_{TV}, \tag{A.11}
\]

where \( v_1 \) and \( v_2 \) are the first and second components of the optical flow \( v \), \( \mu \) is a multiplier factor to weight the strength of the regularizer and the weighted isotropic TV norm is defined by

\[
\| f \|_{TV(D)} = \int_D \sqrt{(g_1(x) \nabla_x f(x))^2 + (g_2(x) \nabla_y f(x))^2} dx,
\]

where \( g_1 \) and \( g_2 \) are given by

\[
g_1(x) = \exp(-\zeta \| \nabla_x I(x) \|_2) + \nu, \tag{A.12}
\]

\[
g_2(x) = \exp(-\zeta \| \nabla_y I(x) \|_2) + \nu. \tag{A.13}
\]

where \( \nu \) is small constant, preventing \( g_1 \) and \( g_2 \) to take the value 0 and \( \zeta \) is a normalizing factor. TV is desirable in the context of occlusion detection because it does not

\[
\nabla I(x, t) \doteq \begin{bmatrix} I \left( x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, t \right) - I(x, t) \end{bmatrix}^T
\]

\[
I_t(x, t) \doteq I(x, t + dt) - I(x, t). \tag{A.8}
\]
penalize motion discontinuities significantly. The overall problem can then be written as the minimization of the cost functional $\psi = \psi_{\text{data}} + \psi_{\text{reg}}$, which is

$$\hat{v}_1, \hat{v}_2, \hat{e}_1 = \arg \min_{v_1, v_2, e_1} \|\nabla I v + I_t - e_1\|_{L^2(D)}^2 + \lambda \|e_1\|_{L^0(D)} + \mu \|v_1\|_{TV(D)} + \mu \|v_2\|_{TV(D)} \quad (A.14)$$

In a digital image, the domain $D$ is quantized into an $M \times N$ lattice $\Lambda$, so we can write (A.14) in matrix form as:

$$\hat{v}_1, \hat{v}_2, \hat{e}_1 = \arg \min_{v_1, v_2, e_1} \frac{1}{2} \|A[v_1, v_2, e_1]^T + b\|_{\ell^2}^2 + \lambda \|e_1\|_{\ell_0} + \mu \|v_1\|_{TV} + \mu \|v_2\|_{TV} \quad (A.15)$$

where $e_1 \in \mathbb{R}^{MN}$ is the vector obtained from stacking the values of $e_1(x, t)$ on the lattice $\Lambda$ on top of one another (column-wise), and similarly with the vector field components $\{v_1(x, t)\}_{x \in \Lambda}$ and $\{v_2(x, t)\}_{x \in \Lambda}$ stacked into $MN$-dimensional vectors $v_1, v_2 \in \mathbb{R}^{MN}$. The spatial derivative matrix $A$ is given by

$$A = [\text{diag}(\nabla_x I) \ \text{diag}(\nabla_y I) \ - I],$$

where $I$ is the $MN \times MN$ identity matrix, and the temporal derivative $\{I_t(x, t)\}_{x \in \Lambda}$ is stacked into $b$. For finite-dimensional vectors $u \in \mathbb{R}^{MN}$, $\|u\|_{\ell^2} = \sqrt{\langle u, u \rangle}$, $\|u\|_{\ell_0} = |\{u_i \neq 0\}|$ and $\|u\|_{TV}$ is defined as

$$\|u\|_{TV} = \sum \sqrt{(g_1)_i(u_{i+1} - u_i)^2 + (g_2)_i(u_{i+M} - u_i)^2}$$

where $g_1$ and $g_2$ are the stacked versions of $\{g_1(x)\}_{x \in \Lambda}$ and $\{g_2(x)\}_{x \in \Lambda}$.

The problem (A.15) is NP-hard when solved with respect to the variable $e_1$ whose nonzero elements indicates the occluded region at each pixel in the image. A straightforward relaxation into a convex would simply replace the $\ell_0$ norm with $\ell_1$. Unfortunately, this implies that “bright” occluded regions are penalized more than “dim” ones.
which is clearly not desirable. Therefore, we relax the $\ell_0$ norm with the weighted-$\ell_1$ norm such that

$$\hat{v}_1, \hat{v}_2, \hat{e}_1 = \arg \min_{v_1, v_2, e_1} \frac{1}{2} \| A[v_1, v_2, e_1]^T + b \|_2^2 + \lambda \| W e_1 \|_{\ell_1} + \mu \| v_1 \|_{TV} + \mu \| v_2 \|_{TV}.$$ (A.16)

where $W$ is a diagonal matrix and resort to an iterative procedure called reweighted-$\ell_1$, proposed by Candès et al. [CWB08] to adapt the weights so as to better approximate the $\ell_0$ norm. $W$ is initially set to be the identity matrix, and correspondingly (A.16) is the customary convex relaxation\(^6\) of the original NP-hard problem [Tib96]. Each iteration has a globally optimal solution that can be reached efficiently from any initial condition. An improved approximation of the $l_0$ norm can be obtained by adapting the weight $W$ iteratively, for instance choosing $W$ to be a diagonal with elements $w(x) \approx 1/(|e_1(x)| + \epsilon)$ as proposed in [CWB08]. The resulting solution of (A.16) greatly improves sparsity, and the residual $e_1$ is closer to a piecewise constant (indicator) function, as shown in Fig. A.1.

Note that the residual $e_1$ in (A.5) is sometimes referred to as modeling *illumination changes* [SH89, Neg98, TLC05, KMK05]. However, even though the model (A.5) appears similar, the *priors* on $e_1$ are rather different. They favor smooth illumination changes; we favor sparse occlusions. While sparsity follows directly from the assumption (i), illumination changes would require a *reflectance function* to be modeled. Instead, all models of the form (A.5) lump reflectance and illumination into a single irradiance term [SYJ03].

---

\(^6\)This norm has been previously used in optical flow estimation, and it makes sense in that context where occlusions are the “nuisance factors.” In our context, however, occlusions are the object of inference, and we do not wish to suppress them in order to provide an optical flow reading in the occluded region, where it is undefined. Instead, optical flow is the nuisance. Therefore, while interesting, this interpretation offers no insight. Instead, we prefer using the reweighted approach as a better approximation of the original problem (A.16) that does not penalize bright occlusions.
## A.2 Minimization with Nesterov’s Algorithm

In this section, we describe an efficient algorithm to solve (A.16) based on Nesterov’s first order scheme [Nes83] which provides $O(1/k^2)$ convergence in $k$ iterations, whereas for standard gradient descent, it is $O(1/k)$, a considerable advantage for a large scale problem such as (A.16). To simplify the notation we let $(e_1)_i \doteq w_i (e_1)_i$, so that $A \doteq [\text{diag}(\nabla_x I) \; \text{diag}(\nabla_y I) \; -W^{-1}]$. The main steps of the algorithm are shown in the following table.

<table>
<thead>
<tr>
<th>Initialize $v_1^0, v_2^0, e_1^0$. For $k \geq 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Compute $\nabla \psi(v_1^k, v_2^k, e_1^k)$</td>
</tr>
<tr>
<td>2. Compute $\alpha_k$ and $\tau_k$</td>
</tr>
<tr>
<td>$\alpha_k = 1/2(k + 1), \tau_k = 2/(k + 3)$</td>
</tr>
<tr>
<td>3. Compute $y_k$</td>
</tr>
<tr>
<td>$y_k = [v_1^k, v_2^k, e_1^k]^T - (1/L)\nabla \psi(v_1^k, v_2^k, e_1^k)$,</td>
</tr>
<tr>
<td>4. Compute $z_k$</td>
</tr>
<tr>
<td>$z_k = [v_1^0, v_2^0, e_1^0]^T - (1/L) \sum_{i=0}^{k} \alpha_i \nabla \psi(v_1^i, v_2^i, e_1^i)$,</td>
</tr>
<tr>
<td>5. Update $[v_1^k, v_2^k, e_1^k]^T = \tau_k z_k + (1 - \tau_k) y_k$.</td>
</tr>
</tbody>
</table>

Stop when the solution converges.

In order to implement this scheme, we need to address the nonsmooth nature of $\ell_1$. This is done in [Nes05], that has already been used profitably for noise reduction, inpainting and deblurring [WBA09, DHJ09], and incorporated in software libraries for
sparse recovery [BBC09]. In our case, we write \( \psi(v_1, v_2, e_1) \) as a summation of terms
\[
\psi(v_1, v_2, e_1) = \psi_1(v_1, v_2, e_1) + \lambda \psi_2(e_1) + \mu \psi_3(v_1) + \mu \psi_4(v_2), \tag{A.17}
\]
and compute the gradient of each term separately: The first is straightforward
\[
\nabla_{v_1, v_2, e_1} \psi_1(v_1, v_2, e_1) = A^T A [v_1, v_2, e_1]^T + A^T b.
\]
The other three, however, require smoothing. \( \psi_2(e_1) = \|e_1\|_1 \) can be rewritten in terms of its conjugate \( \psi_2(e_1) = \max_{\|u\|_\infty \leq 1} \langle u, e_1 \rangle \). The smooth approximation proposed in [Nes05] is
\[
\psi_2^\sigma(e_1) = \max_{\|u\|_\infty \leq 1} \langle u, e_1 \rangle - \frac{1}{2} \sigma \|u\|_2^2
\]
which is differentiable; its gradient is \( u^\sigma \), the optimal solution of (A.18). Consequently, \( \nabla_{e_1} \psi_2^\sigma(e_1) \) is given by
\[
u_i^\sigma = \begin{cases} 
\sigma^{-1}(e_1)_i, & \|(e_1)_i\| < \sigma, \\
\text{sgn}((e_1)_i), & \text{otherwise}.
\end{cases}
\]
Following the lines of [BBC09], \( \nabla_{v_1} \psi_3 \) is given by is given by
\[
\nabla_{v_1} \psi_3^\sigma(v_1) = G^T u^\sigma
\]
where \( G = [G_1, G_2]^T \), \( G_1 \) and \( G_2 \) are weighted horizontal and vertical differentiation operators, and \( u^\sigma \) has the form \([u^1, u^2]^T\) where
\[
u_{i}^{1,2} = \begin{cases} 
\sigma^{-1}(G_{1,2}v_1)_i, & \|[(G_1v_1)_i (G_2v_1)_i]^T\|_{\ell_2} < \sigma, \\
\|[(G_1v_1)_i (G_2v_1)_i]^T\|_{\ell_2}^{-1}(G_{1,2}v_1)_i, & \text{otherwise}.
\end{cases}
\]
\( \nabla_{v_2} \psi_4 \) can also be computed in the same way. We now have all the terms necessary to compute
\[
\nabla \psi(v_1, v_2, e_1) = \nabla \psi_1 + [\lambda \nabla_{e_1} \psi_2, \mu \nabla_{v_1} \psi_3, \mu \nabla_{v_2} \psi_4]^T
\]
\[
\text{(A.22)}
\]
We also need the Lipschitz constant $L$ to compute the auxiliary variables $y_k$ and $z_k$ to minimize $\psi$. Since $\|G^T G\|_2$ is bounded above [DHJ09] by 8, given the coefficients $\lambda$ and $\mu$, $L$ is given by

$$L = \max(\lambda, 8\mu)/\sigma + \|A^T A\|_2.$$  

A crucial element of the scheme is the selection of $\sigma$. It trades off accuracy and speed of convergence. A large $\sigma$ yields a smooth solution, which is undesirable when minimizing the $\ell_1$ norm. A small $\sigma$ causes slow convergence. We have chosen $\sigma$ empirically, although the continuation algorithm proposed in [BBC09] could be employed to adapt $\sigma$ during convergence.

### A.3 Minimization with Split-Bregman

We also describe an alternative method for solving the optimization problem (A.16) based on the split-Bregman method proposed by Goldstein and Osher [GO09], by decoupling the differentiable and non-differentiable portions of the cost function (A.16).

We replace $G_x v_{(1,2)}$ by $d_x^{(1,2)}$ and $G_y v_{(1,2)}$ by $d_y^{(1,2)}$ yielding to a constrained problem,

$$\hat{v}_1, \hat{v}_2, \hat{e}_1 = \arg min_{v_1, v_2, e_1} \frac{1}{2\mu} \|A[v_1, v_2, e_1]^T I_t\|_{\ell_2}^2 + \frac{\lambda}{\mu} \|W e_1\|_{\ell_1} + \|d_x^{(1)}\|_{\ell_1} + \|d_y^{(1)}\|_{\ell_1}$$

subject to:

$$d_x^{(1)} = G_x v_1, \quad d_y^{(1)} = G_y v_1,$$

$$d_x^{(2)} = G_x v_2, \quad d_y^{(2)} = G_y v_2.$$  

where for finite dimensional vectors $u_1, u_2 \in \mathbb{R}^{MN}$,

$$\|\{u_1, u_2\}\|_{\ell_1} = \sum_{i=1}^{MN} \sqrt{(u_1)_i^2 + (u_2)_i^2}.$$  

(A.24)
By relaxing the hard constraints, the cost function (A.23) takes a form that can be minimized by split-Bregman such that

\[
\hat{v}_1, \hat{v}_2, \hat{e}_1, \hat{d}_{(x,y)}^{(1,2)} = \arg\min_{v_1, v_2, e_1, d_{(x,y)}^{(1,2)}} \frac{1}{2\mu} \| A[v_1, v_2, e_1]^T + I_{\ell_2} \|^2
\]

\[
+ \frac{\lambda}{\mu} \| W e_1 \|_{\ell_1} + \| (d_x^1, d_y^1) \|_{\ell_1} + \| (d_x^2, d_y^2) \|_{\ell_1}
\]

\[
+ \frac{\beta}{2} \| d_x^1 - G_x v_1 - b_x^1 \|_{\ell_2}^2 + \frac{\beta}{2} \| d_y^1 - G_y v_1 - b_y^1 \|_{\ell_2}^2
\]

\[
+ \frac{\beta}{2} \| d_x^2 - G_x v_2 - b_x^2 \|_{\ell_2}^2 + \frac{\beta}{2} \| d_y^2 - G_y v_2 - b_y^2 \|_{\ell_2}^2
\]

(A.25)

where \(\beta\) indicates the amount of relaxation. To solve (A.25), we divide the optimization problem into three subproblems and solve them iteratively. The first subproblem is

\[
\hat{v}_1^{k+1}, \hat{v}_2^{k+1} = \arg\min_{v_1, v_2} \frac{1}{2\mu} \| A[v_1, v_2, e_1^{k+1}]^T + I_{\ell_2} \|^2
\]

\[
+ \frac{\beta}{2} \| (d_x^1)^{k+1} - G_x v_1 - (b_x^1)^{k+1} \|_{\ell_2}^2 + \frac{\beta}{2} \| (d_y^1)^{k+1} - G_y v_1 - (b_y^1)^{k+1} \|_{\ell_2}^2
\]

\[
+ \frac{\beta}{2} \| (d_x^2)^{k+1} - G_x v_2 - (b_x^2)^{k+1} \|_{\ell_2}^2 + \frac{\beta}{2} \| (d_y^2)^{k+1} - G_y v_2 - (b_y^2)^{k+1} \|_{\ell_2}^2
\]

(A.26)

The solution of this problem is straightforward. From the optimality conditions, we reach to the following system of equations

\[
\left( \frac{1}{\mu} I_x^T I_x + \beta (G_x^T G_x + G_y^T G_y) \right) v_1 + \frac{1}{\mu} I_x^T I_y v_2 = - \frac{1}{\mu} I_x^T (-e_1 + I_t) + \beta \left( G_x^T (d_x^1 - b_x^1) + G_y^T (d_y^1 - b_y^1) \right),
\]

\[
\left( \frac{1}{\mu} I_y^T I_y + \beta (G_x^T G_x + G_y^T G_y) \right) v_2 + \frac{1}{\mu} I_y^T I_x v_1 = - \frac{1}{\mu} I_y^T (-e_1 + I_t) + \beta \left( G_x^T (d_x^2 - b_x^2) + G_y^T (d_y^2 - b_y^2) \right).
\]

where to simplify the notation we have defined the diagonal matrices \(I_x = \text{diag}(\nabla_x I)\) and \(I_y = \text{diag}(\nabla_x I)\). Following [GO09], to achieve efficiency, we solve the system of
equations using Gauss-Seidel’s method. The component-wise Gauss-Seidel solution to this problem is given by

\[
\begin{align*}
(v_1)_i &= \frac{-\mu(k_2)_i (I_y)_i (I_x)_i + (k_1)_i (\mu(I_y)_i^2 + (k_3)_i)}{(k_3)_i(\mu(I_x)_i^2 + \mu(I_y)_i^2 + (k_3)_i)} = G^1_i \\
(v_2)_i &= \frac{-\mu(k_1)_i (I_y)_i (I_x)_i + (k_2)_i (\mu(I_x)_i^2 + (k_3)_i)}{(k_3)_i(\mu(I_x)_i^2 + \mu(I_y)_i^2 + (k_3)_i)} = G^2_i
\end{align*}
\]

where \(k_1, k_2, \text{ and } k_3\) are given by

\[
(k_1)_i = -\mu(I_x)_i (-e_1)_i + (I_1)_i \\
+ \beta \left((G^T_x G_x v_1)_i + (G^T_y G_y v_1)_i\right) \\
+ \beta \left((G^T_x d^1_x)_i + (G^T_y d^1_y)_i + (G^T_x b^1_x)_i + (G^T_y b^1_y)_i\right)
\]

\[
(k_2)_i = -\mu(I_y)_i (-e_1)_i + (I_1)_i \\
+ \beta \left((G^T_x G_x v_2)_i + \beta (G^T_y G_y v_2)_i\right) \\
+ \beta \left((G^T_x d^2_x)_i + (G^T_y d^2_y)_i + (G^T_x b^2_x)_i + (G^T_y b^2_y)_i\right)
\]

\[
k_3 = \beta \text{ diag}(G^T_x G_x + G^T_y G_y).
\]

Subsequently, we need to solve the second subproblem which is given by

\[
\left(\hat{d}^{(1,2)}_x\right)^{k+1}, \left(\hat{d}^{(1,2)}_y\right)^{k+1} = \arg\min_{d^{(1,2)}_x, d^{(1,2)}_y} \| (d^{(1,2)}_x, d^{(1,2)}_y) \|_{\ell_1}
\]

\[
+ \frac{\beta}{2} \| (d^{(1,2)}_x) - G_x v^{(1,2)} - (b^{(1,2)}_x)^k \|_{\ell_2}^2 \\
+ \frac{\beta}{2} \| (d^{(1,2)}_y) - G_y v^{(1,2)} - (b^{(1,2)}_y)^k \|_{\ell_2}^2.
\]

This problem can be solved analytically using the generalized shrinkage formula, proposed by Wang et al. [WYZ07], such that

\[
\left(\hat{d}^{(1,2)}_x\right)^{k+1} = \max(s^k - 1/\beta, 0) \cdot \frac{G_x v^{(1,2)}_x + (b^{(1,2)}_x)^k}{s^k}
\]

\[
\left(\hat{d}^{(1,2)}_y\right)^{k+1} = \max(s^k - 1/\beta, 0) \cdot \frac{G_y v^{(1,2)}_y + (b^{(1,2)}_y)^k}{s^k}
\]
where \( s_{(1,2)}^k \) is given by
\[
(s^k)_i = \sqrt{|G_x v^k_{(1,2)} + (b_{x}^{(1,2)})^k|^2 + |G_y v^k_{(1,2)} + (b_{y}^{(1,2)})^k|^2}.
\]

The remaining subproblem is
\[
\hat{e}_1 = \argmin_{e_1} \frac{1}{2\mu} \| A[v_1, v_2, e_1]^T + I_t \|_{\ell_2}^2 + \frac{\lambda}{\mu} \| We_1 \|_{\ell_1}.
\] (A.30)

and can also be solved using shrinkage operator. The solution is
\[
(e_1)^{k+1}_i = \frac{r^k_i}{|r^k_i|} \max(|r^k_i| - \lambda w_i, 0).
\] (A.31)

where
\[
r^k = I_x v^k_1 + I_y v^k_2 + I_t.
\]

The main steps of the algorithm can be summarized as follows

\begin{tabular}{|l|}
\hline
\textbf{Initialize} \( v_1, v_2, e_1, d_{x}^{1}, d_{x}^{2}, d_{y}^{1}, d_{y}^{2} \) and \( d_{y}^{2} \) with 0. For \( k \geq 0 \)
\begin{align*}
v_1^{k+1} &= G_1^k, \quad v_2^{k+1} = G_2^k \\
(d_{x}^{(1,2)})^{k+1} &= \max(s^k - 1/\beta, 0) \frac{G_x v^k_{(1,2)} + (b_{x}^{(1,2)})^k}{s^k}
\end{align*}
\begin{align*}
(d_{y}^{(1,2)})^{k+1} &= \max(s^k - 1/\beta, 0) \frac{G_y v^k_{(1,2)} + (b_{y}^{(1,2)})^k}{s^k}
\end{align*}
\begin{align*}
(b_{x}^{(1,2)})^{k+1} &= (b_{x}^{(1,2)})^k + \left( G_x v^{k+1}_{(1,2)} - (d_{x}^{(1,2)})^{k+1} \right)
\end{align*}
\begin{align*}
(b_{y}^{(1,2)})^{k+1} &= (b_{y}^{(1,2)})^k + \left( G_y v^{k+1}_{(1,2)} - (d_{y}^{(1,2)})^{k+1} \right)
\end{align*}
\begin{align*}
(e_1)^{k+1}_i &= \frac{r^k_i}{|r^k_i|} \max(|r^k_i| - \lambda w_i, 0)
\end{align*}
\hline
\end{tabular}

\textbf{Stop} when the solution converges.

**A.4 Experiments**

Following Section A.0.2, evaluation of our algorithm on public available standard datasets is not straightforward, because these typically do not provide ground-truth occlusions. The only benchmark that provides occlusion, in at least parts of the dataset,
is [BSL07], so we used it as a starting point, and generated occlusion maps as follows: for each training sequence, we computed the residual given the ground truth motion and marked the regions where the residual is high. Next, we annotated the regions where ground truth is not defined. Finally, we manually fixed obvious errors in the occlusion maps. In this section, we evaluate the motion estimation and occlusion detection performance of our approach on this dataset and on the well-known Flower Garden sequence. We have also compared our algorithm to [WPZ08], [BA96] and [KZ01] quantitatively.

To handle the large motion, we run our method on a Gaussian pyramid with a scale factor 0.5 up to 5 levels. We also apply 5 warping steps at each pyramid level. In all the experiments, the coefficient $\lambda$ is fixed at 0.01 while $\mu$ is increased gradually from 0.00008 to 0.01 with each warping step at each pyramid level. Relying less on the prior of the flow field at the early warping steps results in more accurate flow estimates. For the re-weighting step, we have also fixed the coefficient $\epsilon$ to 0.001. In our experiments, we also use a non-linear pre-filtering of the images to reduce the influence of illumination changes [ROF92, WPZ08, SRB10] to initialize the re-weighting stage with an accurate flow field. However, at re-weighting steps we use the original images since pre-filtering reduces the occlusion detection accuracy.

We start with unit weights $W = \mathcal{I}$ and solve the convex problem (A.16) (referred as Huber-$\ell_1$ model in our experiments). We then adapt the weights iteratively, thus improving sparsity and achieving a better approximation of the indicator function $e_1$ of the occluded domain, Fig. A.1. One can also observe a gradual improvement of the sparsity of $|We|$ after each re-weighting iteration, Fig. A.2. At each step, the accuracy of occlusion detection also improves.

Representative results for the Flower Garden sequence are shown in Fig. A.3, where the complex occlusions produced by the foliage are also detected successfully.
Figure A.1: The result of the proposed approach on “Venus” from [BSL07] and “Flower Garden.” The first column shows the motion estimates, color-coded as in [BSL07], the second is the residual $I(x,t) - I(w(x), t+dt)$ before re-weighting stage; the third shows $|W e_1|$ after re-weighting, and the fourth is the sparse error term $e_1$.

Figure A.2: This figure illustrates the initial estimate of the error term $e_1$ (first column) and how sparsity of $|W e_1|$ improves with each of the three re-weighting iterations.

In Fig. A.5 and Fig. A.6 we show the effects of re-weighting on the Middlebury data set. The weighted $e_1$ is not only sparser compared to the residual $|I(x,t) -$
Figure A.3: Occlusion and motion estimates for more frames of the Flower Garden. Left to right: initial frame, flow estimate (left), initial estimate of the error term $e_1$ (middle), and occluded region (right).

$I(w(x), t + dt)$ computed before the re-weighting steps but also has a superior occlusion detection accuracy unlike the residual which contains regions that are not occluded. One might think that the residual could just be thresholded, instead of iteratively re-weighted. To evaluate that, we have generated precision-recall curves and observed the change of occlusion detection performance in terms of F-measure by thresholding both signals while varying the threshold value in the interval $[0, 1]$, Fig. A.4. In most cases, the accuracy of the re-weighting approach is superior and more stable under the varying threshold values since $|W e_1|$ better approximates an indicator function. Therefore, one can just choose non-zero elements of $e_1$ to detect occluded regions instead of searching for a global threshold. Note that here the weight

\[\text{Notice that } w(x) > 0, \forall x. \text{ Therefore, } W e_1 \neq 0 \iff e_1 \neq 0\]
matrix $W$ is the one computed at the previous re-weighting step. We have also observed that the precision-recall curves for $|Wc_1|$ does not span the whole recall range, since recall value 1 is not reachable unless all the zero-elements added to the decision which is not meaningful for the analysis of a sparse signal (Fig. A.4, PR-curves).

We have also compared our approach to the robust flow estimation methods proposed by Black and Anandan [BA96] (using the improved version (Classical-L) by Sun et al. [SRB10]) and Wedel et al. [WPZ08] by evaluating the occlusion detection accuracy on the residual $|I(x, t) - I(w(x), t + dt)|$ computed using their flow fields, Fig. A.4. Furthermore, since violations of the symmetry between backward and forward flow estimations have been used intensively as an indicator of occlusions [ADP07, IK08, KZ01, SLK05], we have also implemented another baseline algorithm checking such mismatch between flow fields estimated with [WPZ08]. Note that neither residual nor flow field symmetry violations provides a sparse solution, therefore, to reach to a conclusion, it is required to threshold such a signal after normalization. Hence, we have measured the accuracy of these techniques in terms of precision-recall and F-measure under varying threshold, Fig. A.4. Our method outperforms both approaches.

One might be tempted to regularize the geometry of the occluded region, for instance by adding a regularizing term $\|Wc_1\|_{TV}$ to (A.16). We have also evaluated this model, Fig. A.4. However, occlusions can manifest themselves with very complex geometry and topology, as the Hydrangea in Fig. A.6 and Fig. A.4 illustrate. In such cases, a geometric regularizer is counter-productive as it generates a large number of missed detections.

We have compared our occlusion detection results to [KZ01], using the code provided on-line by the authors. Table A.1 shows that we outperform [KZ01]. Comparing motion estimates gives an unfair advantage to our algorithm because their approach is
based on quantized disparity values, so the accuracy of our motion estimates is predictably superior.

Table A.1: A comparison of the F-measure of our algorithm and [KZ01] on the Middlebury dataset. Since Kolmogorov et al. [KZ01] provide an occlusion detector whose output is binary, we simply compute the precision and recall of the output and report the F-measure based on these values. For comparison, we chose non zero elements of $e_1$ as detected occlusions and provide F-measure with respect to them.

<table>
<thead>
<tr>
<th></th>
<th>Venus</th>
<th>RubberWhale</th>
<th>Hydrangea</th>
<th>Grove2</th>
<th>Grove3</th>
<th>Urban2</th>
<th>Urban3</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-measure [KZ01]</td>
<td>0.63</td>
<td>0.28</td>
<td>0.31</td>
<td>0.62</td>
<td>0.52</td>
<td>0.43</td>
<td>0.53</td>
</tr>
<tr>
<td>F-measure (our method)</td>
<td><strong>0.77</strong></td>
<td><strong>0.52</strong></td>
<td><strong>0.37</strong></td>
<td><strong>0.67</strong></td>
<td><strong>0.60</strong></td>
<td><strong>0.69</strong></td>
<td><strong>0.83</strong></td>
</tr>
</tbody>
</table>

We have also compared the accuracy of the solution of Nesterov’s algorithm and split-Bregman’s method and their convergence speed, Fig. A.5, Fig. A.6, Fig. A.4, Table A.2 and Table A.3. Both methods provide similar performance both in occlusion detection and motion estimation. However, split-Bregman method converges significantly faster, Table A.2.

We have evaluated the accuracy of the flow estimates of our method and compared to other robust flow estimation techniques [BA96, SRB10, WPZ08], Table A.3. Huber-$\ell_1$-TV model minimized with Nesterov’s algorithm provides superior accuracy. However, once the re-weighting stage is initialized with these estimates, and flow estimation is performed on the original images instead of the pre-filtered ones, the accuracy decreases.

Finally, we have evaluated the performance of our algorithm on the Middlebury evaluation dataset, Table-A.4. Nesterov’s algorithm is used to estimate the optical flow on the evaluation set. The flow fields estimated before the reweighting stage are
Table A.2: The comparison of convergence time of the split-Bregman method and Nesterov’s algorithm.

<table>
<thead>
<tr>
<th></th>
<th>Venus</th>
<th>RubberWhale</th>
<th>Hydrangea</th>
<th>Grove2</th>
<th>Grove3</th>
<th>Urban2</th>
<th>Urban3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nesterov’s method</td>
<td>222 secs</td>
<td>342 secs</td>
<td>355 secs</td>
<td>463 secs</td>
<td>494 secs</td>
<td>499 secs</td>
<td>483 secs</td>
</tr>
<tr>
<td>Split Bregman method</td>
<td>90 secs</td>
<td>111 secs</td>
<td>133 secs</td>
<td>260 secs</td>
<td>360 secs</td>
<td>288 secs</td>
<td>277 secs</td>
</tr>
</tbody>
</table>

Table A.3: Quantitative comparison of the proposed models and other robust flow estimation methods [BA96, SRB10, WPZ08] in terms of Average Angular Error (AAE) / Average End Point Error (AEPE)

<table>
<thead>
<tr>
<th>Model</th>
<th>Venus</th>
<th>RubberWhale</th>
<th>Hydrangea</th>
<th>Grove2</th>
<th>Grove3</th>
<th>Urban2</th>
<th>Urban3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huber-$\ell_1$-TV (Nesterov)</td>
<td>3.99/0.28</td>
<td>2.94/0.09</td>
<td>2.09/0.17</td>
<td>2.19/0.15</td>
<td>6.78/0.67</td>
<td>2.59/0.29</td>
<td>4.35/0.66</td>
</tr>
<tr>
<td>$\ell_2$-reweighted-$\ell_1$-TV (Nesterov)</td>
<td>3.96/0.31</td>
<td>5.09/0.16</td>
<td>2.36/0.19</td>
<td>2.60/0.17</td>
<td>7.71/0.78</td>
<td>3.41/0.38</td>
<td>4.91/0.76</td>
</tr>
<tr>
<td>$\ell_2$-reweighted-$\ell_1$-TV (split-Bregman)</td>
<td>4.09/0.33</td>
<td>4.91/0.16</td>
<td>2.34/0.19</td>
<td>2.31/0.15</td>
<td>7.72/0.75</td>
<td>2.86/0.35</td>
<td>4.20/0.63</td>
</tr>
<tr>
<td>Black &amp; Anandan [BA96]</td>
<td>7.81/0.44</td>
<td>5.06/0.14</td>
<td>2.48/0.21</td>
<td>2.76/0.20</td>
<td>6.90/0.75</td>
<td>4.06/0.54</td>
<td>11.18/0.94</td>
</tr>
<tr>
<td>Classic-L [SRB10]</td>
<td>4.75/0.29</td>
<td>3.15/0.09</td>
<td>2.06/0.17</td>
<td>2.49/0.17</td>
<td>6.49/0.66</td>
<td>2.96/0.37</td>
<td>4.72/0.60</td>
</tr>
<tr>
<td>Wedel et al. (Improved L1-TV) [WPZ08]</td>
<td>4.45/0.30</td>
<td>3.61/0.11</td>
<td>2.25/0.18</td>
<td>3.26/0.23</td>
<td>7.07/0.69</td>
<td>2.74/0.36</td>
<td>6.26/0.64</td>
</tr>
</tbody>
</table>

ranked $10^{th}$ and $12^{th}$ in terms of AEPE and AAE respectively while the estimates after reweighting step are ranked $13^{th}$ in terms of both error measures as of June 7, 2011. The optical flow estimation accuracy of our algorithm on the evaluation set is presented in Table A.4. Our method also detects the occluded regions accurately on most of the sequences at evaluation set, Fig. A.7.

A.5 Discussion

We have presented an algorithm to detect occlusions and establish correspondence between two images. It leverages on a formulation that, starting from standard as-
Table A.4: Quantitative evaluation of the proposed models on Middlebury test set in terms of AAE/AEPE.

<table>
<thead>
<tr>
<th>w/o reweighting</th>
<th>Army</th>
<th>Mequon</th>
<th>Schefflera</th>
<th>Wooden</th>
<th>Grove</th>
<th>Urban</th>
<th>Yosemite</th>
<th>Teddy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.62/0.09</td>
<td>2.92/0.16</td>
<td>4.49/0.22</td>
<td>3.14/0.17</td>
<td>3.26/0.75</td>
<td>3.52/0.54</td>
<td>5.10/0.22</td>
<td>2.02/0.53</td>
</tr>
<tr>
<td>w reweighting</td>
<td>4.30/0.11</td>
<td>4.03/0.30</td>
<td>5.13/0.42</td>
<td>3.56/0.20</td>
<td>3.23/0.75</td>
<td>3.22/0.33</td>
<td>2.81/0.15</td>
<td>1.94/0.55</td>
</tr>
</tbody>
</table>

assumptions (Lambertian reflection, constant diffuse illumination), arrives at a variational optimization problem. We have shown how this problem can be relaxed into a sequence of convex optimization schemes, each having a globally optimal solution, and presented two efficient numerical schemes for solving it.

We emphasize that our approach does not assume a rigid scene, or a single moving object. It also does not assume that the occluded region is simply connected. Instead, our model is general under the assumptions (a)-(b), and allows arbitrary (piece-wise diffeomorphic) domain deformations, corresponding to an arbitrary number of moving or deforming objects, and an arbitrary number of simply connected occluded regions (jointly represented by a multiply-connected domain $\Omega$).

The fact that occlusion detection reduces to a two-phase segmentation of the domain into occluded $\Omega$ and visible region $D \setminus \Omega$ should not confuse the reader familiar with the image segmentation literature whereby two-phase segmentation of one object (foreground) from the background can be posed as a convex optimization problem [CEN06]. Note that in the approach of [CEN06] the problem can be made convex only in occluded region term, $e_1$, but not jointly in both $e_1$ and the motion field, $v$. Therefore, such an approach does not in general yield a global minimum.

The limitations of our approach stand mostly in its dependency from the regularization coefficients $\lambda, \mu$ and coefficient $\sigma$ in the optimization. In the absence of some
estimate of the variance coefficient $\lambda$, one is left with painstakingly tuning it by trial-and-error. Similarly, $\mu$ is a parameter that, like in any classification problem, trades off missed detections and false alarms, and therefore no single value is “optimal” in any meaningful sense. These limitations are shared by most variational optical flow estimation algorithms.
Figure A.4: Comparison of the occlusion detection accuracy of different variants of proposed technique and two other baseline methods in terms of precision-recall curves and F-measure. The first baseline method considers the residual of the optical flow estimated via [BA96] or [WPZ08] as an indicator of occlusions while the second one identifies the regions of mismatch in forward and backward flow fields estimated with [WPZ08] as occluded.
Figure A.5: This figure presents the occlusion and motion estimates on the sequences Venus, Grove2 and Grove3 from Middlebury dataset. The odd rows, left to right: ground truth optical flow, flow estimates before re-weighting stage and flow estimates after reweighing with Nesterov’s algorithm and split-Bregman method. The even rows, left to right: ground truth occluded regions, the initial estimate of the error term \( e_1 \), the estimate of \( |W e_1| \) after the reweighting step with Nesterov’s algorithm and split-Bregman method.
Figure A.6: This figure presents the occlusion and motion estimates on the sequences RubberWhale, Hydrangea, and Urban2 from Middlebury dataset. Each sequence occupies two rows. The odd rows, left to right: ground truth optical flow, flow estimates before re-weighting stage and flow estimates after reweighing with Nesterov’s algorithm and split-Bregman method. The even rows, left to right: ground truth occluded regions, the initial estimate of the error term $e_1$, the estimate of $|W e_1|$ after the reweighting step with Nesterov’s algorithm and split-Bregman method.
Figure A.7: Occlusion and flow estimates on the sequences Teddy, Grove, Wooden, Yosemite, Backyard, Basketball, Dumptruck, and Evergreen from Middlebury evaluation set: Estimated flow fields are depicted at odd rows while the estimates of $|W_{e1}|$ after the reweighting step are illustrated at even ones.
REFERENCES


[R] R&D division of Santa Monica Studios. FutureLight.


