ACTION RECOGNITION USING LOG-COVARIANCE MATRICES OF SILHOUETTE AND OPTICAL-FLOW FEATURES

KAI GUO

Dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

ACTION RECOGNITION USING LOG-COVARIANCE MATRICES OF SILHOUETTE AND OPTICAL-FLOW FEATURES

by

KAI GUO

B.S., Xidian University, 2005,
MPhil., City University of Hong Kong, 2007

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2012

Approved by

First Reader

Prakash Ishwar, Ph.D.
Associate Professor of Electrical and Computer Engineering

Second Reader

Janusz Konrad, Ph.D.
Professor of Electrical and Computer Engineering

Third Reader

W. Clem Karl, Ph.D.
Professor of Electrical and Computer Engineering

Fourth Reader

Stan Sclaroff, Ph.D.
Professor of Computer Science

Fifth Reader

Fatih Porikli, Ph.D.
Senior Principal Research Scientist
Mitsubishi Electric Research Laboratories

Nothing in life is to be feared, it is only to be understood.
Now is the time to understand more,
so that we may fear less. Marie Curie

Acknowledgments

I would like to thank my advisors, Prof. Janusz Konrad and Prof. Prakash Ishwar, for introducing me to video processing and analytics. I am very grateful to them for providing me with an opportunity to explore research challenges and for guiding me through the research labyrinth. I want also to thank them for the moral and financial support, and the confidence bestowed on me.

I also want to thank Prof. W. Clem Karl from the ECE department, Prof. Stan Sclaroff from the CS department and Dr. Fatih Porikli from Mitsubishi Electric Research Laboratory for taking time to serve on my doctoral committee and for giving me valuable feedback on my dissertation. I would also like thank Dr. Porikli for sharing with me his expertise in covariance matrix representations. Additionally, I would like to thank Prof. Pierre Moulin from the University of Illinois Urbana-Champaign for suggesting the log-Euclidean metric.

I would also like to thank all my friends, especially my colleagues in the Information Sciences and Systems group for their friendship and support during these four years in Boston.

My final gratitude goes to my parents and my wife Yao Yu. There are no words in the world that could fully express my appreciation for their love and support. I love you.

ACTION RECOGNITION USING LOG-COVARIANCE MATRICES OF SILHOUETTE AND OPTICAL-FLOW FEATURES

(Order No.)

KAI GUO

Boston University, College of Engineering, 2012

Major Professors: Prakash Ishwar, Ph.D., Associate Professor of Electrical and Computer Engineering
Janusz Konrad, Ph.D., Professor of Electrical and Computer Engineering

ABSTRACT

Algorithms for recognizing human actions in a video sequence are needed in applications such as video surveillance and video search and retrieval. Developing algorithms that are not only accurate but also efficient is challenging due to the complexity of the task and the sheer size of video.

In this thesis, we develop a general framework for compactly representing, quickly comparing, and accurately recognizing actions using empirical covariance matrices of features. With each pixel we associate a feature vector which provides a localized description of the action. This generates a spatio-temporally dense collection of action feature vectors. We use the empirical covariance matrix of this feature vector collection as a low-dimensional representation of the action. We use two supervised learning methods, the nearest-neighbor classification and sparse-linear approximation classification, for action recognition using labeled training dictionaries of action co-

variance matrices. Common to both methods is the novel idea that classification algorithms that have been developed for vectors can be re-purposed for covariance tensors by using a log-nonlinearity to map the convex cone of covariance matrices to the vector space of symmetric matrices.

We illustrate the approach on two types of action feature vectors. One is based on silhouette tunnels of moving objects, and the other is based on optical flow. Action feature vectors of the first type describe the shape of the silhouette tunnel. Action feature vectors of the second type describe various motion characteristics such as velocity, gradient, and divergence. We demonstrate state-of-the-art recognition performance for both types of action feature vectors on the Weizmann, KTH, YouTube and the low-resolution ICPR-2010 challenge data sets under modest CPU requirements.

We also demonstrate how our approach can be used for sequentially detecting changes in actions in an adaptive, unsupervised manner so as to parse a long video into sub-videos, each containing only a single action class. We use a non-parametric statistical framework to learn the distribution of the nearest-neighbor Riemannian distances between feature covariance matrices of video segments. Then, we use binary hypothesis testing to determine if new video segments include action changes. Our algorithm can detect 98.36% of action boundaries with 0.19% false alarm rate.

We conclude by discussing how our framework can be adapted to recognize human interactions, which is usually a more challenging problem due to occlusion between moving individuals. We develop an approach based on dividing human interactions into separate sequences, each containing a single individual, and then combining the estimated action likelihoods for each individual sequence.

The excellent performance of log-covariance-matrix representation combined with sparse-linear approximation classification demonstrated here for action recognition should encourage the use of this framework for other recognition problems.

Contents

List of Figures

List of Abbreviations

BoG	.....	Bag of Features
CCR	.....	Correct Classification Rate
DPN	.....	Dynamic Probabilistic Network
FOV	.....	Field of View
GMM	.....	Gaussian Mixture Model
HDP	.....	Hierarchical Dirichlet Processes
HMM	.....	Hidden Markov Model
LDA	.....	Latent Dirichlet Allocation
LSI	.....	Latent Semantic Indexing
LSK	.....	Local Steering Kernel
MCS	.....	Matrix Cosine Similarity
MEI	.....	Motion Energy Image
MHI	.....	Motion History Image
MIL	.....	Multiple Instance Learning
MRF	.....	Markov Random Field
NN	.....	Nearest Neighbor
PCA	.....	Principle Component Analysis
SIFT	.....	Scale-invariant Feature Transform
SLA	.....	Sparse Linear Approximation
SVM	.....	Support Vector Machine

Chapter 1

Introduction

The proliferation of network cameras in the last decade has led to surveillance video overload. The volume and complexity of generated visual information far exceeds the capacity of human operators to manage, analyze and respond in real-time. It is, therefore, critical to develop efficient and effective automatic methods to analyze video data with the goal of understanding the visual environment. Of the many facets of video analysis, action recognition stands out as particularly important, since it can lead to many practical applications in such areas as video surveillance, video search and retrieval and human-computer interaction.

This thesis concentrates on the problem of action recognition, which is defined as follows: given a dictionary of annotated training action videos, recognize the unknown action of a query video. The problem is illustrated in Fig. 1·1. We would like to recognize the action "walking" based on the prior knowledge of several actions.

This chapter contains a general introduction to the problem and motivation for this thesis. First, we discuss some practical issues of action recognition. Some interesting applications follow. Finally, we describe our contributions and present outline of the thesis.

1.1 Action recognition challenges

Despite significant efforts by the computer vision and image processing communities, recognizing actions in video is still a challenging problem on account of the following

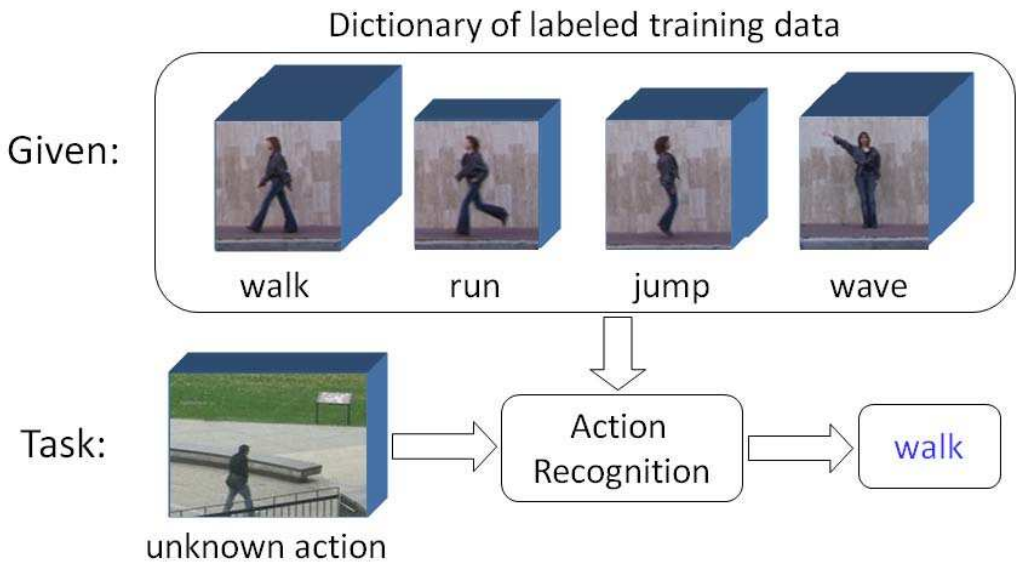

Figure 1·1: Illustration of action recognition: input is a query video with an unknown action, output is an action label of this query video.

issues:

- Scene complexity: many videos are captured under uncontrolled conditions that may involve clutter and occlusions, as shown in Fig. 1·2. Given a video with a number of persons performing various activities, it is hard to track each individual and recognize his/her action. Occlusions, by hiding part of a moving object, may also hinder effectiveness of action recognition, if discriminative information of an action is missing from the video.

- Acquisition complexity: in many situations, a camera's field of view (FOV) is not fixed during video recording. A camera which is panned, tilted or zoomed-in/-out can make a truly static background in real life appear to be "moving" in the video. Although humans can easily distinguish a truly moving foreground from a "moving" background based on experience, computers are not intelligent enough to do so. Therefore, action recognition may be severely affected by non-stationary background in videos. Another acquisition complexity results from

the dependence of object appearance on viewpoint. An action captured from different viewpoints may look significantly different, as shown in Fig. 1·3.

- Action complexity: there exists intra-class motion variability and inter-class motion ambiguity. No two individuals perform the same action in exactly the same manner. On the other hand, different actions may look similar in some particular poses, such as jumping and skipping.

Figure 1·2: Example of a video frame captured under uncontrolled conditions that involves clutter and occlusions.

Figure 1·3: The action "walking" looks quite different from different viewpoints.

Due to the practical issues discussed above, the focus of this research is on the key sub-problem in which the video footage only involves actions from a single moving object without significant occlusions. Such footage may be obtained by detecting, tracking and isolating object trajectories, although in some scenarios it may be non-trivial. In addition to the three sources of complexity described above, the volume of video data is an issue. Discriminative representations that are amenable to rapid processing (to enable close to real-time operation) and have a low storage cost are needed.

1.2 Applications

Action recognition can lead to interesting applications in many areas, such as:

- video surveillance,
- sports and entertainment,
- wildlife monitoring,
- human computer interaction.

We discuss some of these applications in detail below.

1.2.1 Video surveillance

Video surveillance is an important tool to enhance public safety and privacy protection. It has long been deployed in public places such as airports, train stations and city centers. An example of surveillance cameras and monitors is shown in Fig. 1·4. Millions of cameras have been mounted in large cities, such as London, Chicago and New York. Those cameras produce endless video streams, while only a tiny fraction of them can be processed. Since most of the surveillance videos are not critical, one just wants to discover unusual actions from videos. If each video camera embeds an

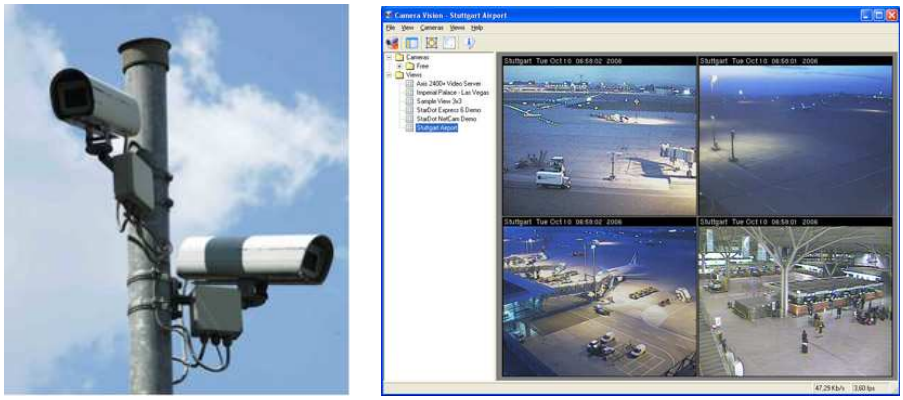

Figure 1·4: Surveillance cameras and monitors.

algorithm that can distinguish unusual actions and only transmits the relevant videos, it can save a great deal of bandwidth and human labor.

1.2.2 Sports and entertainment

Figure 1·5: Examples of sports and entertainment that can use action recognition.

Action recognition may be a useful tool to understand the content of sports and entertainment videos (Fig. 1·5). Today, with the popularity of video websites (YouTube, MegaVideo, etc.), it is quite easy to share sports and entertainment videos with friends and family. Sometimes we are interested in searching videos that are similar to our

own videos. Currently, this is based on metadata matching. However, labeling videos to produce metadata is time-consuming and tedious. Actions are usually the most important information in videos, and therefore a better video retrieval technique may be based on action matching. A new video search engine can be developed where input is a query video and output is its matched video that has most similar action to that in the query video.

1.2.3 Wildlife monitoring

Figure 1·6: Wildlife monitoring example.

Action recognition includes human actions as well as animal actions. Researchers who study wildlife behavior usually cannot stay in animal habitats for a long time. Also, the presence of a human may affect animals' behavior. Video cameras can replace humans and help better understand wildlife behavior. An intelligent camera can be triggered to record videos when an animal's movement is detected. Based on action recognition, perhaps different animals can be even distinguished.

1.2.4 Human-computer interaction

Action recognition can also be used for human-computer interaction. Microsoft has developed a sensor called *Kinect* that can provide "controller-free gaming and entertainment experience" (Fig. 1·7). It enables users to control and interact with Xbox

Figure 1·7: Kinect sensor and human-computer interaction.

360 without using a physical controller but instead by using gestures and voice. Perhaps in the coming years, a new Windows operating system will have a more natural integrated user interface instead of traditional keyboard and mouse interface.

1.3 Thesis goals and contributions

Our goals in the thesis can be summarized as follows:

- develop a systematic framework for action recognition, including action representation and classification;
- propose an action change detection algorithm;
- study the human interaction recognition problem.

1.3.1 Action recognition framework

The accuracy and efficiency of an action recognition method critically depends on:

- how to model and represent actions;
- how to classify actions.

In the thesis, we develop a novel method for action representation, based on covariance matrix of a bag of local features. Covariance matrix is a compact representation

since it extracts the second order statistics of local features and lies in a much lower dimensional space than local features themselves. The selected features need to include discriminative properties for action classification. We implemented two types of local features, those based on silhouette tunnel and those based on optical flow. Silhouette tunnel describes the shape of an action and optical flow describes the motion dynamics of an action. Both of them include important characteristics for classifying human actions.

Action recognition can be considered as a supervised learning problem, in which we can determine the query action class based on the dictionary of labeled action samples. In this thesis, we recognize an action using two classifiers: the nearest neighbor (NN) classifier and the sparse linear approximation (SLA) classifier. The NN classifier has been widely used in many supervised learning problems since it is simple, effective and training free. The SLA was proposed by Wright *et. al* to recognize human faces (Wright et al., 2009). The classification is based on sparse linear approximation of a query sample with respect to an overcomplete dictionary of training samples (base elements).

1.3.2 Action change detection

Our action recognition framework assumes that each video sequence includes only one action class, which is not the case in many practical scenarios. Therefore, we need to parse a long video sequence into sub-sequences, each of which only includes a single action class. In this context, we develop a method that can identify time instants when an action change happens (e.g., from walking to running). In other words, we partition a video into many sub-videos so that each of them contains only one single action. We use the non-parametric statistical framework to learn the distribution of the nearest-neighbor distance between covariance matrices of video segments. Then, we use binary hypothesis testing to examine if a new video segment includes action

change. It is an unsupervised learning method that can discover temporal boundaries of actions without knowing action labels.

1.3.3 Human interaction recognition

We also study how to recognize human interactions, such as kissing, handshaking, hugging, etc. It is a more challenging problem because:

- partial occlusions often happen between moving individuals;
- individual actions may not be sufficiently informative to help us fully understand the interaction.

In this thesis, given an interaction video, we firstly divide it into individual actions using object tracking. Then, we estimate a confidence measure for each individual that reflects the likelihood of each action class. Finally, we combine the confidence measure to obtain the interaction label.

1.3.4 Summary of main contributions

The main contributions of our work are summarized as follows.

- We develop a new action recognition framework based on log-covariance matrices of bags of local action features;
- We generalize the vector-based NN and SLA Classification algorithms to covariance matrices;
- We discover discriminative features for action recognition from object silhouette tunnels and optical flow;
- We extend the feature covariance matrix framework to the unsupervised action change detection problem;

- We develop a feature covariance based method for human interaction recognition.

1.4 Outline of the thesis

The outline of the thesis is as follows:

Chapter 2 reviews related work on action recognition. It categorizes past work with respect to action representation model and classification algorithm used.

Chapter 3 introduces our proposed action recognition framework, including action representation and classification. Log-covariance matrices of features are used to represent actions. Both NN and SLA classifiers are employed for action classification. In NN classifier, two distance metrics for covariance matrices (affine-invariant Riemannian metric and log-Euclidean metric) are used instead of the Euclidean distance.

Chapter 4 discusses two examples of action feature vectors, one based on silhouette tunnels and the other based on optical flow.

Chapter 5 discusses practical implementation issues for our action recognition framework, i.e., how to process continuous video with limited memory, how to deal with temporal misalignment between training and query segments, etc.

Chapter 6 shows experimental results for our action recognition framework, based on four benchmark datasets: Weizmann, KTH, UT-Tower and YouTube. This chapter also tests the robustness of our method to action variability and viewpoints, followed by computational complexity analysis.

Chapter 7 proposes an algorithm to detect action changes in a video, based on a non-parametric framework and binary hypothesis testing. Experimental results are provided to show the effectiveness of our method.

Chapter 8 is dedicated to human interaction recognition. The basic idea is to separate interaction into individuals, analyze each individual's action and finally

obtain the interaction label by considering the most likely concurrence of individual actions.

Chapter 9 discusses contributions of this thesis, draws conclusions and presents possible directions for future work.

Chapter 2

Related Work

In this chapter we review related work in action recognition. There are basically two important issues in action recognition: action representation and action classification. Accordingly, the existing literature on action recognition, which is vast and rapidly growing, can be loosely grouped by the type of model used to represent actions and the type of classification algorithm used.

2.1 Action representation

There are roughly five types of features for action representation that have been studied in the literature: shape-based features, motion-based features, geometric human body features, interest-point features and dynamic models. These are discussed in more detail in the following sections.

2.1.1 Shape-based features

Shape-based action representation is widely used in action recognition today. A partial listing of papers using shape models includes (Bobick and Davis, 2001; Yilmaz and Shah, 2005; Gorelick et al., 2007; Chen et al., 2008; Guo et al., 2009; Wang et al., 2007; Ikizler and Duygulu, 2007; Zhang et al., 2009; Ahmad et al., 2010). This representation relies on accurately capturing the silhouette of a moving object at each time instant. A sequence of such silhouettes forms a silhouette tunnel, i.e., a spatio-temporal binary mask of the moving object changing its shape in time. A silhouette

tunnel is desirable for action recognition as it is invariant to luminance, color, and texture of the moving object as well as the background. Although silhouette tunnels do not capture motion inside objects, the moving silhouette boundary leaves a very distinct signature of occurring activity.

An effective method based on silhouette tunnels was developed by Gorelick *et al.* (Gorelick et al., 2007). At each pixel, the expected length of a random walk to the silhouette tunnel boundary, which can be computed by solving a Poisson equation, is treated as a shape feature of the silhouette tunnel. An action classification based on this approach was shown to be remarkably accurate suggesting that the method is capable of extracting highly-discriminative information. Collins and Gross (Collins et al., 2002) have also used silhouettes to identify human actions, but their method extracts key frames from the query video sequence and matches them with key frames from training videos. The classification is performed by the nearest-neighbor rule based on normalized correlation scores. This method is conceptually simple and easy to implement, but it is based on 2-D silhouettes without considering the dynamics of silhouette evolution. The dynamic nature of video has been also exploited by Bobick and Davis (Bobick and Davis, 2001) who proposed a motion energy image (MEI), that represents where motion has occurred in an image sequence, and motion history image (MHI), that is a scalar field depicting how recently the motion occurred. Together, MEI and MHI act as a two-component version of a temporal template, and are compared with known actions in a database to determine the best action match.

In our work, we use silhouette-based features that fall into the category of shape-based features. We extract various spatial and temporal distances that describe the shape of silhouettes and thus implicitly characterize actions.

2.1.2 Motion-based features

Motion features extract dynamic characteristics of actions, which are some of the most discriminative attributes of actions. A partial listing of papers using motion models includes (Lowe, 2004; Ke et al., 2005; Danafar and Gheissari, 2007; Scovanner et al., 2007; Liu et al., 2008; Fathi and Mori, 2008; Ali and Shah, 2010; Seo and Milanfar, 2011; Wang et al., 2011). Recently, Ali *et al.* (Ali and Shah, 2010) proposed kinematic features derived from optical flow for action representation. Each kinematic feature gives rise to a spatio-temporal pattern. Then, kinematic modes are computed by performing Principle Component Analysis (PCA) on the spatio-temporal volumes of kinematic features. Multiple Instance Learning (MIL) is employed to recognize actions. The idea of MIL is to represent each action video as a collection or a "bag" of kinematic modes in which each kinematic mode is referred to as an instance representing that video. Seo and Milanfar (Seo and Milanfar, 2011) used a 3D local steering kernel (3D-LSK) as an action feature that can reveal global space-time geometric information. The idea behind 3D-LSK is based on analyzing the radiometric (pixel value) differences based on estimated space-time gradients, and using this structure information to determine the shape and size of a canonical kernel (descriptor). They also employed Matrix Cosine Similarity (MCS) as a similarity measure between pairs of activity sequences, which generalizes the notion of cosine similarity between two vectors. Wang *et al.* (Wang et al., 2011) proposed an approach to describe actions by dense trajectories. They sample dense points from each frame and track them based on displacement information from a dense optical flow field. The dense trajectories well characterize actions and a descriptor based on motion boundary histograms is used to encode the trajectory information.

In our work, we also use optical-flow features which fall into the category of motion-based features. The optical-flow features extract dynamic motion attributes

to represent actions. Some of our features have been successfully used for action recognition in Ali *et al.*'s work (Ali and Shah, 2010), such as divergence, vorticity, etc.

2.1.3 Geometric human body features

Geometric human body model can be built with static and dynamic body parameters. They are mostly used in controlled environments where human body parts, such as legs and arms, are easy to identify. A partial listing of papers using geometric human body models includes (Rohr, 1994; Goncalves et al., 1995; Cunado et al., 2003; Wang et al., 2004; Xie et al., 2011). Rohr *et al.* (Rohr, 1994) incorporate a 1 DOF pose parameter to aid in model fitting. All the poses in a walking action are indexed by a single number. The work of Goncalves *et al.* (Goncalves et al., 1995) promotes three-dimensional tracking of the human arm against a uniform background using a two-cone arm model and a single camera. However, acquiring the three-dimensional information from videos is still a very complicated process. Xie *et al.* (Xie et al., 2011) proposed a pose-based approach for locating and recognizing human actions in video using a deformable body part model.

2.1.4 Interest-point features

Interest points, e.g., corners, SIFT features (Lowe, 1999), etc., have also been employed to represent actions. A partial listing of papers using interest-point features includes (Dollar et al., 2005; Smith et al., 2005; Schuldt et al., 2004; Wong and Cipolla, 2007; Niebles et al., 2008; Laptev et al., 2008; Wu et al., 2011; Le et al., 2011; Kovashka and Grauman, 2010). Interest points are sufficiently discriminative and are usually sparse (far fewer interest points than the number of pixels in a video sequence). Niebles (Niebles et al., 2008) and Dollar (Dollar et al., 2005) use 2D Gaussian and 1D Gabor filters to select interest points in a spatio-temporal volume. Laptev

et al. (Laptev et al., 2008) utilize the Harris corner detector to locate local salient pixels with significant local variations in both spatial and temporal domains. Wong *et al.* (Wong and Cipolla, 2007) extract interest points by considering structural information and detecting cuboids in regions that have large probability of containing motion. Wu *et al.* (Wu et al., 2011) recently proposed a new spatio-temporal context distribution feature of interest points for human action recognition. Each action video is expressed as a set of relative XYT coordinates between pairwise interest points in a local region. Then, Gaussian Mixture Model (GMM) was adopted to model the distribution of context features for each video. Le *et al.* (Le et al., 2011) developed an extension of the Independent Subspace Analysis algorithm to learn spatio-temporal features of interest points from unlabeled video data. Their method performs well when combined with deep learning techniques such as staking and convolution to learn hierarchical representations. Kovashka *et al.* (Kovashka and Grauman, 2010) extracted local motion and appearance features, quantized them into visual vocabulary and then formed candidate neighborhoods that are very discriminative for a given action category.

Recently, probabilistic latent semantic analysis and latent Dirichlet allocation have been applied to recognize actions. Distributions of spatio-temporal words and intermediate topics are learned corresponding to human action categories in an unsupervised manner. The algorithms are able to localize multiple actions in complex motion sequences. Wang *et al.* proposed an unsupervised learning framework to model activities and interactions in complicated scenes (Wang et al., 2009) by using hierarchical Bayesian models to connect low-level features, simple "atomic" activities and interactions. Three hierarchical Bayesian models were proposed: the Latent Dirichlet Allocation (LDA) mixture model, the Hierarchical Dirichlet Processes (HDP) mixture model and the Dual Hierarchical Dirichlet Processes (Dual-HDP). Wang and

Mori proposed a new "bag-of-words" representation that characterizes each frame as a single word (Wang and Mori, 2009). Additionally, they use the semi-Latent Dirichlet allocation (S-LDA) model and the probabilistic Latent Semantic Indexing (pLSI) for action recognition.

2.1.5 Dynamic models

Dynamic models are among the earliest models used for human action recognition. A partial listing of papers using dynamic models includes (Yamato et al., 1992; Starner and Pentland, 1995; Raptis et al., 2010). The general idea is to define each static posture of an action as a state, and describe the dynamics (temporal variations) of the action by using a state-space transition model. An action is modeled as a set of states and connections in the state space using a Dynamic Probabilistic Network (DPN). Hidden Markov Model (HMM), the most commonly used DPN, has the advantages of directly modeling time variations of features of data. The parameters of the dynamic model can be learned from a set of training action videos. To classify an action, the joint probability with the maximum value is selected as the criterion for action recognition.

Kale *et al.* (Kale et al., 2004) used an HMM to model human gait. Each individual is characterized with one HMM and several typical silhouettes are used as hidden states for HMM training. Brand *et al.* (Brand and Kettnaker, 2000) proposed a Multi-observation-mixture-counter HMM (MOMC-HMM) to factorize the observation space. Oliver *et al.* further proposed a Coupled HMM (CHMM) (Oliver et al., 2000) to model the temporal and causal correlations among hidden states. Another model, called Dynamically-multi-linked HMM (DML-HMM) was recently developed by Xiang and Gong (Xiang and Gong, 2006). Shi *et al.* (Shi et al., 2008) proposed a semi-Markov discriminative approach by employing a Viterbi-like column generation algorithm.

The models adopted in the literature have their own properties. Shape models are invariant to luminance, color, and texture of the moving object as well as background. The quality of silhouettes greatly affects the success of shape models. Motion models extract the most discriminative attributes of actions, but sometimes background motion can bias the motion models. Geometric human body models can work well in and only in the controlled environment. Interest-point models extract salient features related to moving pixels, but these features are usually sparse and sometimes inadequate to describe an action. Dynamic models can characterize an action very well, but it is complicated and time-consuming to estimate the model parameters. In this thesis, we will focus on shape models and motion models, since they are discriminative for action recognition and they are fairly easy to obtain.

2.2 Action classification

In terms of action classification, algorithms from machine learning community have been heavily borrowed. Some action recognition methods are based on the NN classifier (Gorelick et al., 2007; Bobick and Davis, 2001; Dollar et al., 2005; Cunado et al., 2003; Wang et al., 2004; Seo and Milanfar, 2011; Liu et al., 2008; Lowe, 2004), a straightforward method that requires no explicit training. Other methods recognize actions by using support vector machine (SVM) (Ikizler and Duygulu, 2007; Ahmad et al., 2010; Schuldt et al., 2004; Danafar and Gheissari, 2007; Scovanner et al., 2007; Hoai et al., 2011). The objective of SVM is to maximize the separation margin between classes. The approach uses a kernel function to map the training samples to a high-dimensional feature space and finds an optimal separation hyperplane in this feature space. Recently Hoai *et al.* (Hoai et al., 2011) proposed a method that can jointly perform video segmentation and action recognition using multi-class SVM framework and dynamic programming. Another classification technique used for ac-

tion recognition is boosting (Zhang et al., 2009; Smith et al., 2005; Fathi and Mori, 2008; Ke et al., 2005) which improves performance of any classifier by combining a number of weak classifiers into a strong one. A discussion of various classifiers can be found in (Duda et al., 2001).

In recent years, classification algorithms based on sparse linear representation, which have connections to the area of compressive sensing, have started receiving significant attention in the computer vision and pattern recognition communities (Wright et al., 2009; Mahoor et al., 2011; Mairal et al., 2008; Elad and Aharon, 2006; Yang et al., 2008; Mei and Ling, 2009; Ying et al., 2010; Concha et al., 2010). Successful applications of sparse representation include face recognition (Wright et al., 2009), facial expression recognition (Ying et al., 2010; Mahoor et al., 2011), object recognition (Mairal et al., 2008), image denoising (Elad and Aharon, 2006), super resolution image processing (Yang et al., 2008) and object tracking (Mei and Ling, 2009).

Concha *et al.* (Concha et al., 2010) developed an approach for action recognition based on compressive sensing principles. In their method, feature vectors extracted from each frame are randomly projected into a lower-dimensional space, and the projected features are used in a classification algorithm based on a Hidden Markov Model (HMM). Although compressive sensing and sparse linear representations share a common mathematical foundation, in the work of Concha *et al.*, compressive sensing is mainly used for feature dimensionality reduction rather than classification based on the characteristics of a sparse representation as in (Wright et al., 2009; Mahoor et al., 2011; Mairal et al., 2008; Ying et al., 2010).

In our work, we use the NN classifier and sparse representation classifier for action classification.

Chapter 3

Proposed framework

In the previous chapter, we reviewed state-of-the-art methods for action recognition. In this chapter, we will introduce our proposed framework, including action representation and action classification. Our approach represents an action using covariance matrix of local feature vectors. Then, an action video is classified using supervised learning algorithms: the nearest neighbor classifier or sparse linear approximation (SLA) classifier.

3.1 Action recognition as a supervised learning problem

Action recognition is a supervised learning problem: given annotated training samples and a query sample, the goal is to classify the query sample based on the training set. There are roughly three classes of supervised learning methods: global methods, global methods with local structure, and local methods.

Global methods treat data as abstract vectors living in high-dimensional Euclidean space and train classifiers directly in the high-dimensional representation, as shown in Fig. 3·1. The advantage of global methods is its compatibility with any classifier. Although it takes full consideration of the global structure of data, it ignores local structure that is useful for classification. Moreover, in the action recognition context, few training samples are available but they lie in extremely high-dimensional space (e.g., a $128 \times 128 \times 20$ video clip is approximately vectorized into a 3×10^5-dimensional vector). It is thus unfeasible, in practice, to learn the global statistical characteristics

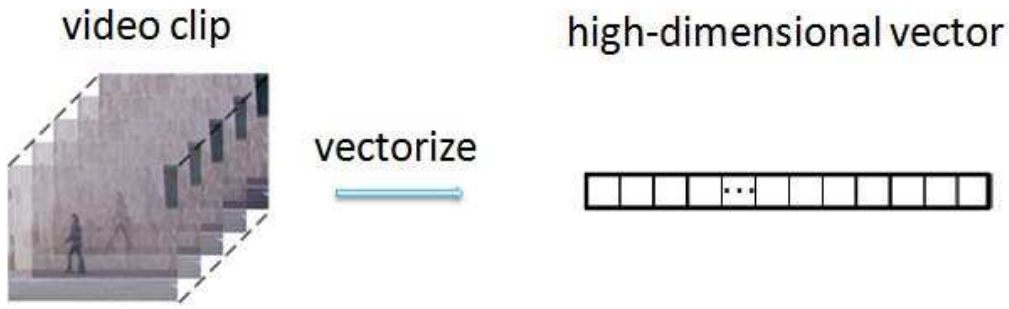

Figure 3·1: Action representation using global methods.

of the high-dimensional vector.

As an alternative to global methods, over the last decade or so, global methods with apriori local structure have emerged, as shown in Fig. 3·2. These methods only

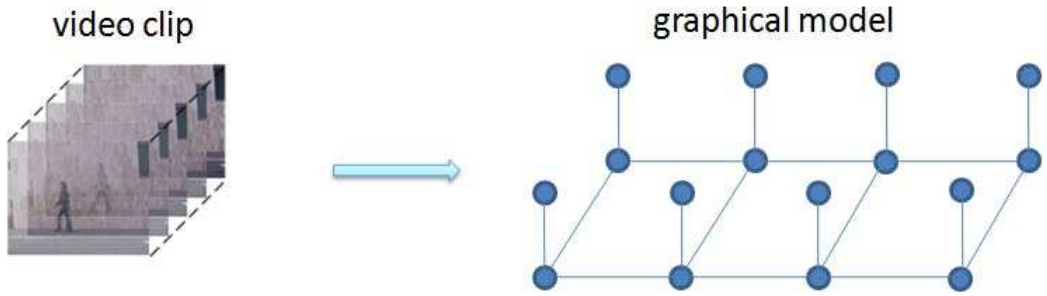

Figure 3·2: Action representation using global methods with local structure.

consider partial correlation within data by using graphical models, e.g., Markov Random Fields (MRF), as an *a priori* local data structure. Such methods are, however, still complex and the parameters of a graphical model are difficult to estimate on the account of very few training samples.

With data dimensionality orders of magnitude larger than the number of training samples, in order to reliably represent an action video, we can extract a *dense* set of localized features from this video, the so-called "bag of local features", as illustrated in Fig. 3·3. The advantage of such a "bag-of-features" action representation is that even from a single training video clip one can extract a very large number of local

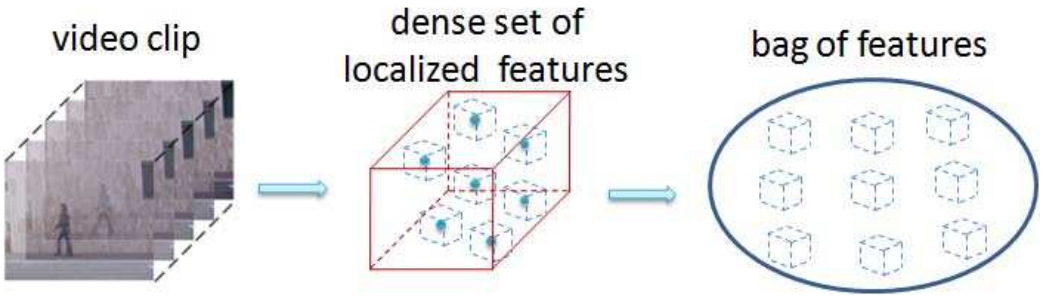

Figure 3·3: Action representation using local methods.

features (one per pixel) from which one can reliably learn the statistical properties of local features. As an example, a single $128 \times 128 \times 20$ training video provides more than 3×10^5 local features. If the local features are sufficiently discriminative, then a high classification accuracy can be realized.

3.2 Action representation

3.2.1 Feature Covariance matrices

One undesirable aspect of the bag of local features is that as a representation its dimensionality is even larger than the video clip from which it was extracted –the number of pixels is multiplied by the size of the feature vector. It is, therefore, desirable to reduce its dimensionality in the feature space. Ideally, one would like to learn the probability density function (pdf) of these local feature vectors. This, however, is not only computationally-intensive, but it may not lead to a lower-dimensional representation: a kernel-based density estimation algorithm needs to, in-effect, store all the samples used to form the estimate. The mean feature-vector, which is low-dimensional, can be learned reliably and rapidly but may not be sufficiently discriminative. This is corroborated by our results in Chapter 6.5, where action recognition based on just the mean feature vector performs poorly. Inspired by Tuzel *et al.*'s work (Tuzel et al., 2006; Tuzel et al., 2008), we discovered that if features are well-

chosen, then the feature-covariance matrix can provide a remarkably discriminative representation for action recognition. The main claim of this thesis is that a feature covariance matrix is "sufficient" for action recognition. In addition to their simplicity and effectiveness, covariance matrices of local features have low storage and processing requirements. Our action representation based on a bag of local features and their low-dimensional covariance matrix is illustrated in Fig. 3·4. For example, 13-dimensional local feature vectors described in Section 4.1 (see Fig. 4·3) lead to a 13×13 covariance matrix that has 91 independent entries due to its symmetry.

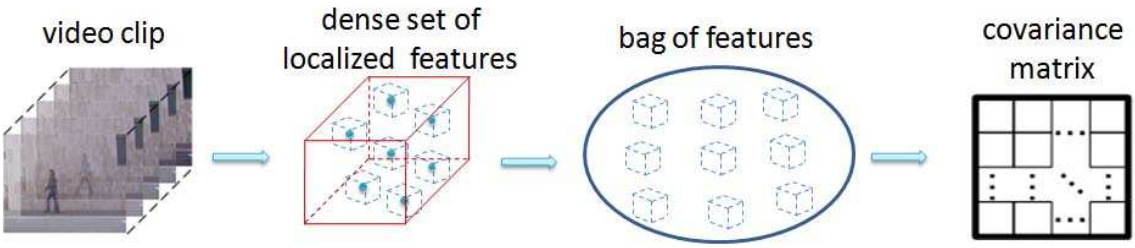

Figure 3·4: Action representation based on a low-dimensional covariance matrix of a bag of local feature vectors.

Let $I(x, y, t)$ denote a video sequence and $\mathcal{F} = \{\mathbf{f}_k\}$ denote a "bag of feature vectors" extracted from the video. Let the size of the feature set $|\mathcal{F}|$ be N. The empirical covariance matrix defined on $\mathcal{F}$ is

$$\mathcal{C} := \frac{1}{N-1} \sum_{k=1}^{N} (\mathbf{f}_k - \mu)(\mathbf{f}_k - \mu)^T, \tag{3.1}$$

where μ is the mean feature vector: $\mu = \sum_{k=1}^{N} \mathbf{f}_k$. The covariance matrix provides a natural way to fuse multiple feature vectors. The dimension of the covariance matrix is only related to the dimension of the feature vectors. If $\mathbf{f}_k$ is d dimensional, then $\mathcal{C}$ is a $d \times d$ matrix. Due to its symmetry, $\mathcal{C}$ has only $(d^2 + d)/2$ independent numbers. Since d is usually much less than N, $\mathcal{C}$ usually lies in a much lower dimensional space than the "bag of feature vectors" that need $N \times d$ dimensions.

3.2.2 Log-covariance matrices

Covariance matrices do not lie in Euclidean space, but instead they form a convex cone. For example, the space of covariance matrices is not closed under multiplication with negative scalers. Most of the common machine learning methods work on Euclidean spaces and thus covariance matrices are not suitable features. One idea to solve this problem is to map the convex cone of covariance matrices to the vector space by using the matrix logarithm, proposed by Arsigny *et al.* (Arsigny et al., 2006), as shown in Fig. 3·5. The log-covariance matrix $L_{\mathcal{S}}$ of a covariance matrix $C_{\mathcal{S}}$ is computed as follows. Suppose that the eigen-decomposition of $C_{\mathcal{S}}$ is given by $C_{\mathcal{S}} = VDV'$, where the columns of V are orthonormal eigenvectors and D is the diagonal matrix of eigenvalues. Then $L_{\mathcal{S}} = \log(C_{\mathcal{S}}) = V\widetilde{D}V'$, where $\tilde{D}$ is a diagonal matrix obtained from D by replacing D's diagonal entries by their logarithms.

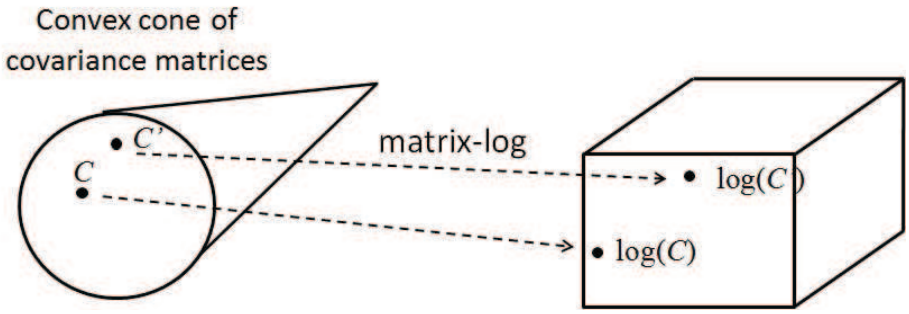

Figure 3·5: Matrix logarithm that maps covariance matrices from a convex cone to Euclidean space.

There are three key ingredients in our action representation method:

- Dense bag of local features,
- Feature covariance matrices,
- Log-covariance matrices.

We define an operator Ψ which transforms an input video clip into an output log-covariance matrix representation, as shown in Fig. 3·6.

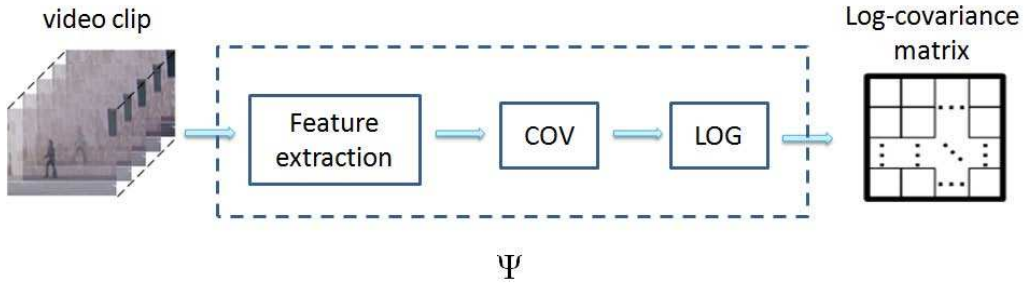

Figure 3·6: Operator Ψ composed of three ingredients in action representation.

3.3 Classification

We have introduced an action representation using low-dimensional log-covariance matrices of a "bag of features". The next problem is how to classify a query action sample based on the representations of training samples. The general framework for action classification is shown in Fig. 3·7. In this thesis, we investigate two learning approaches for action classification: the nearest neighbor classification and sparse linear approximation (SLA) classification.

3.3.1 Nearest-neighbor classification

Nearest-neighbor (NN) classification is one of the most widely used algorithms in supervised learning. The idea is simple and straightforward: given a query sample, find out the most similar sample (under some distance measure) in the annotated training set and assign its label to the query sample as described in Fig. 3·8.

The success of NN classification strongly depends on the distance measure that is used. Therefore, we firstly introduce two appropriate distance metrics for covariance matrices: affine-invariant Riemannian metric and log-Euclidean metric. Then,

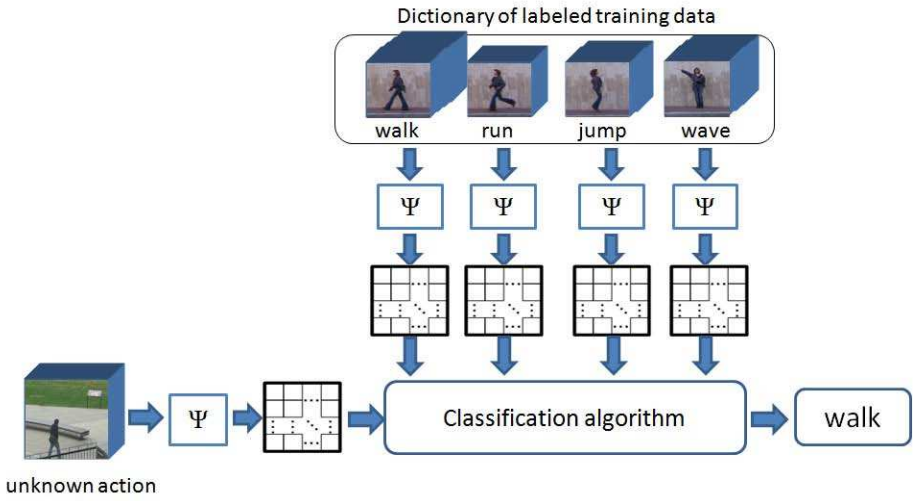

Figure 3·7: Ψ operator that includes the three ingredients in action representation.

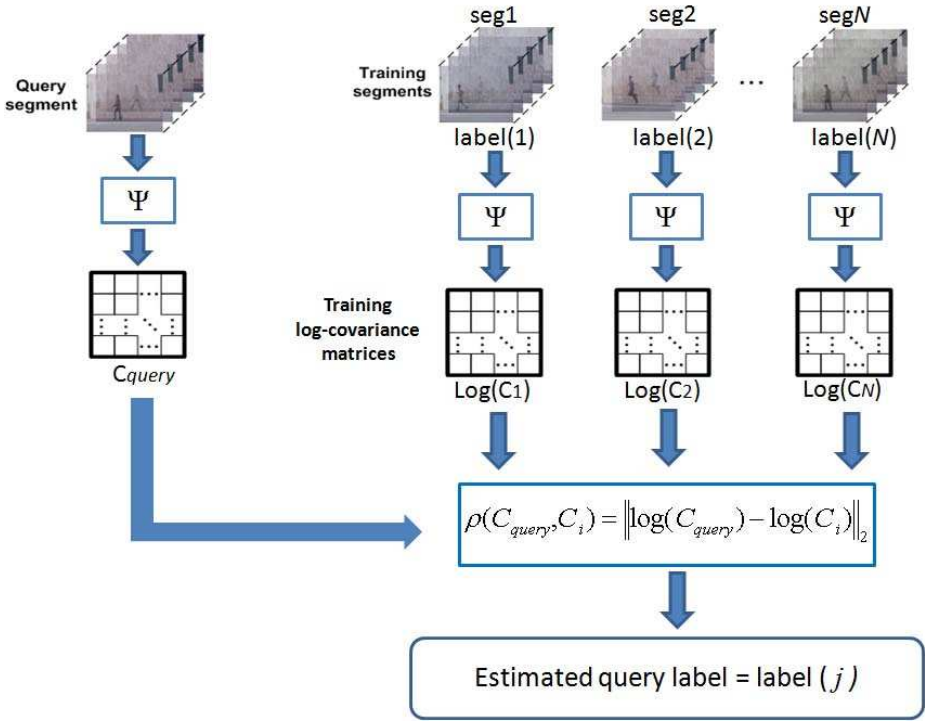

Figure 3·8: Diagram of NN-based classifier.

we analyze the possible reason why log-Euclidean metric outperforms the Euclidean metric in NN classification.

Distance metrics

The set of covariance matrices of a given dimension does not form a vector space (it forms a convex cone). The Euclidean distance does not correctly capture the topology of the convex cone of symmetric non-negative definite matrices. This is corroborated by the experimental results of Chapter 6.5, where a simple NN classification algorithm using the Euclidean metric to measure distances between covariance matrices performs poorly. There are at least two metrics defined on the Riemannian manifold of covariance matrices that have been studied in the literature, namely, the affine-invariant Riemannian metric (Forstner and Moonen, 1999) and the log-Euclidean metric (Arsigny et al., 2006).

A. Affine-invariant Riemannian metric

The affine-invariant Riemannian metric ρ_1 defined below was proposed by Forstner and Moonen (Forstner and Moonen, 1999), and has been successfully used in object tracking and face localization (Tuzel et al., 2006; Tuzel et al., 2008). If C_1 and C_2 are two covariance matrices, then,

$$\rho_1(C_1, C_2) := \quad ||\log(C_2^{-1}C_1)||_2 = \quad \sqrt{\sum_{k=1}^{d} \log^2 \lambda_k(C_1, C_2)}, \tag{3.2}$$

where $\log(\cdot)$ denotes the matrix-logarithm, $||\cdot||_2$ denotes the Frobenius norm on matrices, and $\lambda_k(C_1, C_2)$ are the generalized eigenvalues of C_1 and C_2, i.e.,

$$\lambda_k C_1 \mathbf{u}_k = C_2 \mathbf{u}_k, \tag{3.3}$$

where $\mathbf{u}_k \neq \mathbf{0}$ is the k-th generalized eigenvector. This distance measure captures the manifold structure of covariance matrices and satisfies the metric axioms of positivity, symmetry, and triangle inequality. It is also invariant to invertible affine transformations of the local features, that is, if C_1 and C_2 are respectively the covariance matrices

of $\mathbf{F}_1$ and $\mathbf{F}_2$ and $\bar{C}_1$ and $\bar{C}_2$ are respectively the covariance matrices of $A\mathbf{F}_1 + \mathbf{b}_1$ and $A\mathbf{F}_2 + \mathbf{b}_2$, where A is an invertible matrix, then,

$$\rho_1(C_1, C_2) = \rho_1(\overline{C_1}, \overline{C_2}).$$

B. Log-Euclidean metric

The log-Euclidean metric ρ_2 defined below and proposed by Arsigny *et al.* (Arsigny et al., 2006), is another Riemannian metric on the manifold of covariance matrices. We know that log-covariance matrices lie in a vector space. Then, distances between covariance matrices can be simply measured by any norm (the Frobenius norm in particular) in the transformed Euclidean space, specifically,

$$\rho_2(C_1, C_2) := \quad || \log(C_1) - \log(C_2)||_2 \tag{3.4}$$

where $\log(\cdot)$ denotes matrix-logarithm and $|| \cdot ||$denotes the Frobenius norm on matrices.

3.3.2 Sparse linear classification

In this subsection, we exploit discriminative nature of sparse linear representations (Wright et al., 2009) to perform action classification. This approach is generic and has been applied to many vision tasks, such as face recognition, image super-resolution and image denoising. The key idea underlying this approach is that if the training vectors of all the classes are pooled together and a query vector is expressed as a linear combination of the fewest possible training vectors, then the majority of the training vectors in the linear combination are likely to be of the same class as the query vector. The pooling together of the training vectors of all the classes is important for classification because the training vectors of each individual class may well span the space of all query vectors. The pooling together induces a "competition" among

the training vectors of different classes to approximate the query using the fewest possible training vectors. We extend this approach to action recognition by applying it to (column-wise) vectorized log-covariance matrices that we refer to as samples. Specifically, we approximate the log-covariance matrix of a query segment by a sparse linear combination of log-covariance matrices of all training segments.

If there are sufficiently many training samples, it is likely that the log-covariance vector of the query sample can be well-approximated by a sparse linear combination of the log-covariance vectors of training samples that only come from the category of the query sample. Thus, the sparse coefficients of the linear combination are very informative since the non-zero coefficients, especially large ones, are likely to indicate the label of the query sample. This sparse representation can be obtained by solving an l^1- minimization problem using linear programming. We can then annotate the query sample based on the coefficients of the sparse linear representation. In this section, we first explain how the log-covariance matrix of a query action segment can be approximated by a sparse linear combination of log-covariance matrices of all training action segments by solving an l^1-norm minimization problem. We then discuss how the locations of large non-zero coefficients in the sparse linear approximation can be used to determine the query label. The overall classification framework based on these ideas is shown in Fig. 3·9.

Sparse linear combination

Suppose we have obtained the log-covariance vectors of training samples for each action class: $P_i = [\mathbf{p}_{i,1}, \mathbf{p}_{i,2}, \cdots, \mathbf{p}_{i,n_i}]$, where i denotes the i-th class and n_i is the number of training samples in class i. We can stack up all the training samples column by column to form a matrix $P := [P_1, P_2, \cdots, P_M] \in \mathbb{R}^{K \times N}$, where K is the dimensionality of log-covariance vectors, M is the number of training action classes and $N = \sum_{j=1}^{M} n_j$.

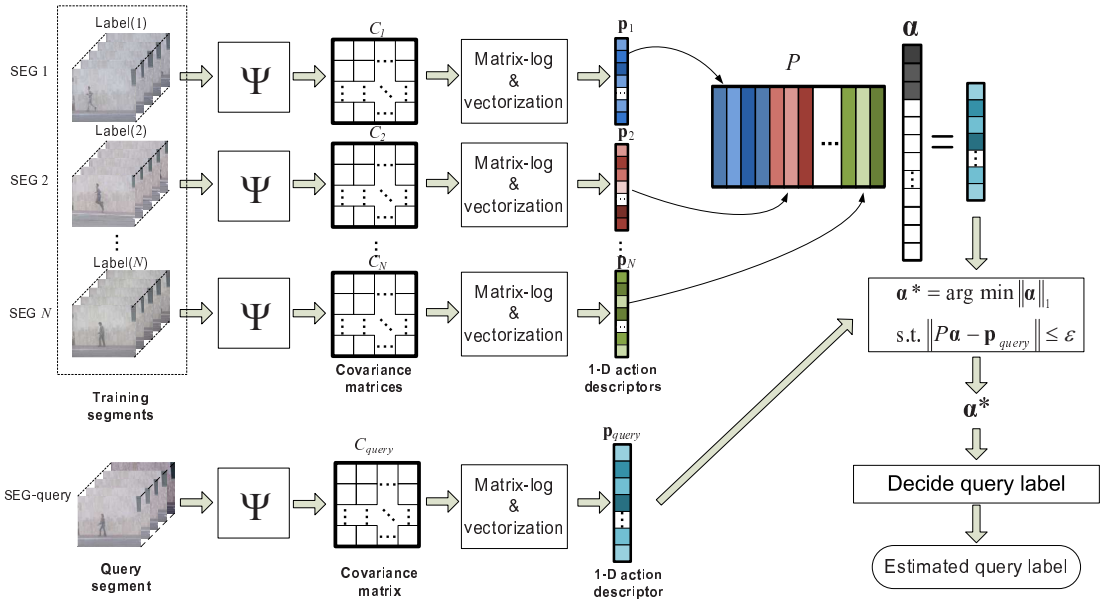

Figure 3·9: Diagram of SLA classification.

Given a query sample $\mathbf{p}_{query}$, one may attempt to express it as a linear combination of training samples by solving the matrix-vector equation given by:

$$\mathbf{p}_{query} = P\boldsymbol{\alpha} \in \mathbb{R}^K, \tag{3.5}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^N$ is the coefficient vector. Since $N \gg K$, the system of linear equations associated with $\mathbf{p}_{query} = P\boldsymbol{\alpha}$ is underdetermined and thus its solution $\boldsymbol{\alpha}$ is not unique. We seek a sparse solution to (3.5) where, under ideal conditions, the only nonzero coefficients in $\boldsymbol{\alpha}$ are those which correspond to the class of the test sample. Such a sparse solution can be found, in principle, by solving the following NP-hard optimization problem:

$$\boldsymbol{\alpha}^* = \arg\min \ \|\boldsymbol{\alpha}\|_0, \qquad s.t. \quad \mathbf{p}_{query} = P\boldsymbol{\alpha}, \tag{3.6}$$

where $\|\cdot\|_0$ denotes the l^0-norm, which counts the number of non-zero entries in a

vector. A key result in the theory of compressive sensing is that if the optimal solution $\boldsymbol{\alpha}^*$ is sufficiently sparse, then solving the l^0-minimization problem (3.6) is equivalent to solving the following l^1-minimization problem

$$\boldsymbol{\alpha}^* = \arg\min \ \|\boldsymbol{\alpha}\|_1, \qquad s.t. \quad \mathbf{p}_{query} = P\boldsymbol{\alpha}. \tag{3.7}$$

This problem is a convex optimization problem that can be solved in polynomial time.

So far we have dealt with the l^1-minimization problem where $\mathbf{p}_{query} = P\boldsymbol{\alpha}$ is assumed to hold exactly. Since action segments may be noisy, it is more accurate to introduce a noise term $\mathbf{z}$ in this system as follows:

$$\mathbf{p}_{query} = P\boldsymbol{\alpha} + \mathbf{z}, \tag{3.8}$$

where $\mathbf{z}$ is an additive noise term whose length is assumed to be bounded by ε, i.e., $\|\mathbf{z}\|_2 \leq \varepsilon$. This leads to the following robust l^1-minimization problem:

$$\boldsymbol{\alpha}^* = \arg\min \ \|\boldsymbol{\alpha}\|_1, \qquad s.t. \quad \|P\boldsymbol{\alpha} - \mathbf{p}_{query}\|_2 \leq \varepsilon. \tag{3.9}$$

SLA-based classification algorithm

We have introduced the SLA framework in which we are interested in finding the sparsest linear approximation $\boldsymbol{\alpha}^*$ of a test sample using training samples from a dictionary. We now describe how to use this framework for action recognition. Each coefficient of $\boldsymbol{\alpha}^*$ weighs the contribution of its corresponding training sample to the representation of the test sample. Ideally, the sparse non-zero coefficients should be only associated with the class of the test sample. However, in practice non-zero coefficients often relate to multiple action classes. Therefore, we need to examine these classes and estimate the class of the test sample. To this end, we follow Wright *et. al*, 2009, and use a measure called the reconstruction residual error (RRE). Let $\boldsymbol{\alpha}_i^* = [\alpha_{i,1}^*, \alpha_{i,2}^*, \cdots, \alpha_{i,n_i}^*]$ denote the coefficients associated with class i, corresponding

to training matrix P_i. The RRE measure of class i is defined as:

$$R_i(\mathbf{p}_{query}) = \|\mathbf{p}_{query} - P_i\boldsymbol{\alpha}_i^*\|_2. \tag{3.10}$$

To annotate the sample $\mathbf{p}_{query}$ we assign the class label that leads to the minimum RRE. The action recognition algorithm based on sparse representation framework can be summarized as follows:

1. Compute the log-covariance descriptor for each action segment;
2. Given $\mathbf{p}_{query}$, solve the l^1-minimization problem (3.9) to obtain $\boldsymbol{\alpha}^*$;
3. Compute RRE for each class i based on (3.10);
4. Annotate the query sample $\mathbf{p}_{query}$ as: $label(\mathbf{p}_{query}) = \arg\min_i R_i(\mathbf{p}_{query})$

Chapter 4

Features

In previous chapter, we introduced a framework for action recognition, including action representation using log-covariance matrices and action classification based on NN or SLA algorithms. This framework is generic and can be applied in different supervised learning scenarios. The success of this framework requires that the selected features represent motion dynamics. In this section, we will introduce two examples of local feature vectors that extract discriminative characteristics of actions in videos.

4.1 Silhouette tunnel shape features

Objects which undergo similar actions can have very different photometric, chromatic and textural properties in different scenes. Motion characteristics are relatively invariant to these properties. One example of features that obey these invariance properties is to base them directly on a sequence of 2-D silhouettes of the moving and deforming object (see Fig. 4·1). Simple background subtraction techniques (Elgammal et al., 2002) and more-advanced spatio-temporal video segmentation methods based on level-sets (Ristivojević and Konrad, 2006) are capable of producing an object silhouette sequence from raw video action sequence. Under ideal conditions, each frame in the silhouette sequence would contain a white mask (white = 1) which exactly coincides with the 2-D silhouette of the moving and deforming object against a "static" black background (black = 0). A sequence of such object silhouettes in time forms a spatio-temporal volume in x-y-t space that we refer to as a silhouette tunnel.

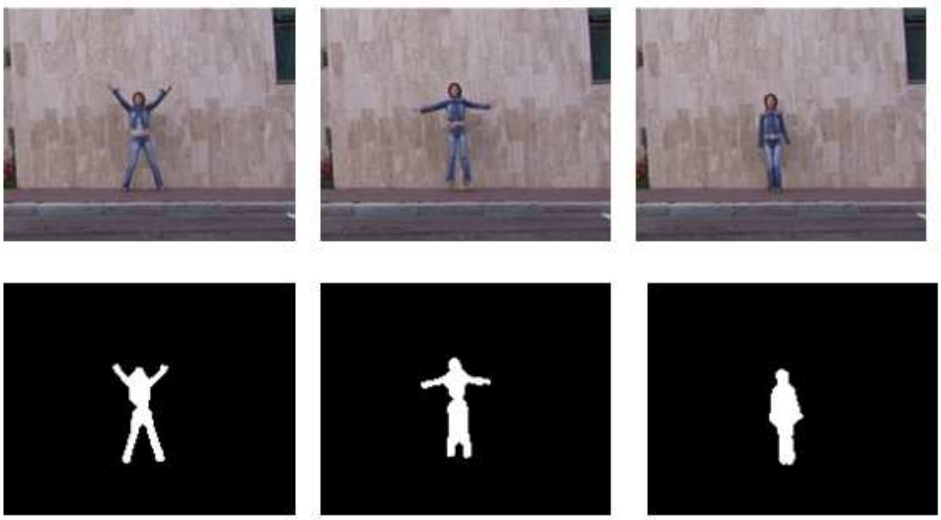

Figure 4·1: Example of a human action sequence from the Weizmann Human Action Database: Three frames from a "jumping-jack" action sequence (top row) and corresponding silhouettes (bottom row).

Since changes in object position are of secondary importance for action recognition, we need to remove object motion. We can do this by aligning the centroids of object silhouettes in the background-subtracted sequence to the same spatial coordinates. Silhouette tunnel actually depicts an action as a 3-D shape. Thus, action comparison is converted into shape comparison, i.e., how to measure the similarity between pairs of 3-D shapes. There is an extensive body of literature devoted to the representation and comparison of shapes of volumetric objects. A variety of approaches have been explored ranging from deterministic mesh models used in the graphics community to statistical models, both parametric (e.g., ellipsoidal models) and non-parametric (e.g., Fourier descriptors). In this dissertation, 3-D silhouette shapes are compared based on the covariance matrices of local feature vectors, extracted from their silhouette tunnels. Fig. 4·2 shows a block diagram of the proposed action representation using silhouette shape features.

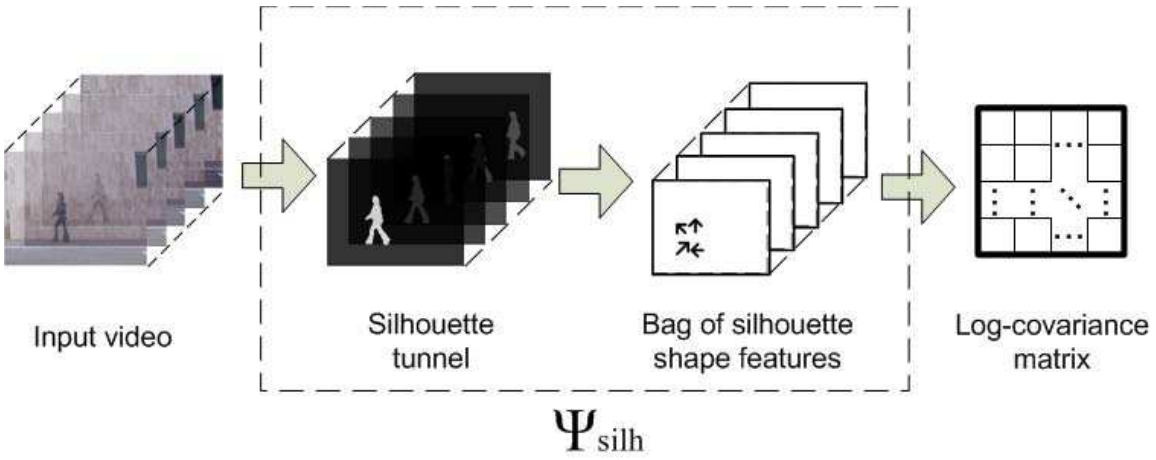

Figure 4·2: Operator Ψ_{silh} that extracts normalized 13×13 covariance matrix of an action in the video.

4.1.1 Shape feature vectors

Let $\mathbf{s} = (x, y, t)^T$ denote the horizontal, vertical, and temporal coordinates of a pixel. Let $\mathcal{A}$ denote the set of coordinates of all pixels belonging to an action segment which is W pixels wide, H pixels tall, and N frames long, i.e., $\mathcal{A} := \{(x, y, t)^T : x \in [1, W], y \in [1, H], t \in [1, T]\}$. Let $\mathcal{S}$ denote the subset of pixel-coordinates in $\mathcal{A}$ which belong to the silhouette tunnel. With each pixel located at $\mathbf{s}$ within the silhouette tunnel, we associate the following 13-dimensional feature vector $\mathbf{f}(\mathbf{s})$ that captures certain shape characteristics of the tunnel (Guo et al., 2009):

$$\mathbf{f}(x, y, t) := [x, y, t, d_E, d_W, d_N, d_S, d_{NE}, d_{SW}, d_{SE}, d_{NW}, d_{T+}, d_{T-}]^T, \tag{4.1}$$

where $(x, y, t)^T \in \mathcal{S}$ and d_E, d_W, d_N, and d_S are Euclidean distances from (x, y, t) to the nearest silhouette boundary point to the right, to the left, above and below the pixel, respectively. Similarly, d_{NE}, d_{SW}, d_{SE}, and d_{NW} are Euclidean distances from (x, y, t) to the nearest silhouette boundary point in the four diagonal directions, while d_{T+} and d_{T-} are similar measurements in the temporal direction. Clearly, these 10 distance measurements capture silhouette tunnel shape as "seen" from location

$(x, y, t)^T$. Fig. 4·3 depicts these features graphically.

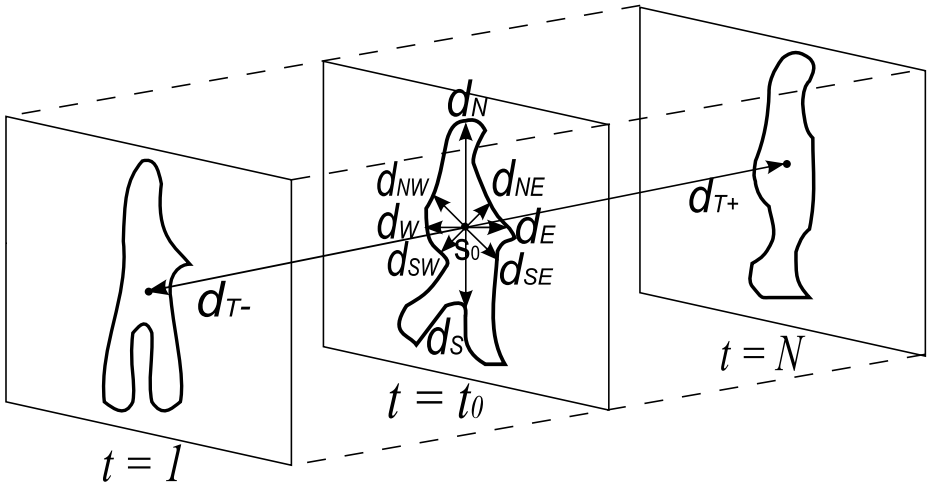

Figure 4·3: Each point $\mathbf{s}_0 = (x_0, y_0, t_0)^T$ of a silhouette tunnel within an N-frame action segment has a 13-dimensional feature vector associated with it: 3 position features x_0, y_0, t_0, and 10 shape features given by distance measurements from (x_0, y_0, t_0) to the tunnel boundary along 10 different spatio-temporal directions shown in the figure.

There is one shape feature vector $\mathbf{f}$ associated with each pixel of a silhouette tunnel, and thus there are a large number of feature vectors. The collection of all feature vectors $\mathcal{F}(\mathcal{S}) := \{\mathbf{f}(\mathbf{s}) : \mathbf{s} \in \mathcal{S}\}$ is an overcomplete representation of the shape of the silhouette tunnel because $\mathcal{S}$ is completely determined by $\mathcal{F}$ and it contains additional data which are redundant. It is instructive to see how individual feature components change with the change of pixel location. Fig. 4·4 depicts each of the 13 features for a single silhouette frame (x-y slice of a silhouette tunnel for a fixed value of t) as an intensity image, where higher brightness means larger value of that feature. In this figure, the origin of the coordinate system is in the left-top corner of the image.

Note that the intensity of the x-component image increases linearly from left to right inside the silhouette whereas the intensity of the y-component image increases from top to bottom. However, the intensity of the t-component image is spatially-

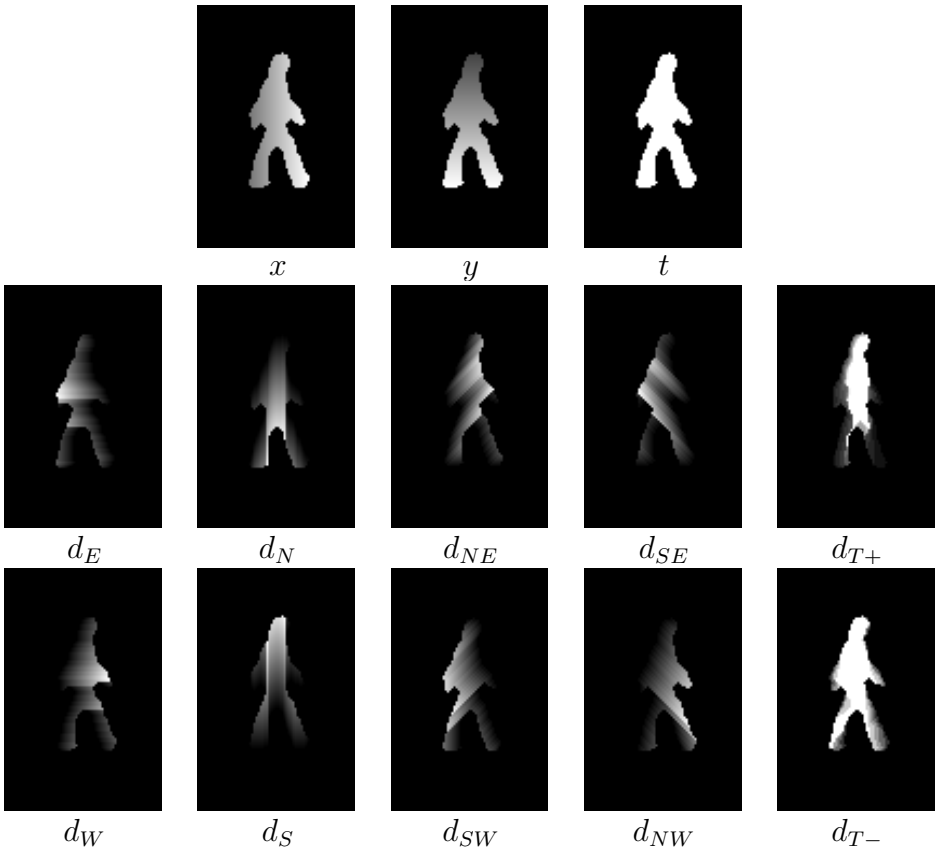

Figure 4·4: Individual components of feature vector $\mathbf{f}(x, y, t)$ depicted as intensity images with t fixed and (x, y) variable. The origin is at the top left corner and brighter points denote larger values.

constant since all pixels in the same frame have the same value of t. Similarly, the d_W image has lower values (dark) at the left of the silhouette since it measures the distance to the left silhouette boundary whereas the d_{T_-} and d_{T_+} images are very bright (large distance) in the torso and darker (shorter distance) within the limb areas. This is to be expected since the position of the torso is largely unchanged across time after centroid alignment whereas legs and arms move significantly. This potentially shortens the temporal distance to the silhouette tunnel boundary.

4.1.2 Shape covariance matrix

After obtaining 13-dimensional silhouette shape feature vectors, we can compute their 13×13 covariance matrix $C_{\mathcal{S}}$, defined as follows. Let $\mathbf{S} = (X, Y, T)^T$ denote a random location vector which is uniformly distributed over $\mathcal{S}$, i.e., the probability mass function of $\mathbf{S}$ is equal to zero for all locations $\mathbf{s} \notin \mathcal{S}$ and is equal to $1/|\mathcal{S}|$ at all locations in $\mathcal{S}$, where $|\mathcal{S}|$ denotes the volume of the silhouette tunnel. Then, $C_{\mathcal{S}} := \text{cov}(\mathbf{F})$ where $\mathbf{F} := \mathbf{f}(\mathbf{S})$. More explicitly,

$$C_{\mathcal{S}} := \text{cov}(\mathbf{F}) = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{S}} (\mathbf{f}(\mathbf{s}) - \boldsymbol{\mu}_F)(\mathbf{f}(\mathbf{s}) - \boldsymbol{\mu}_F)^T \tag{4.2}$$

where $\boldsymbol{\mu}_F = \mathbb{E}[\mathbf{F}] = \sum_{\mathbf{s} \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \mathbf{f}(\mathbf{s})$ is the mean feature vector. Thus, $C_{\mathcal{S}}$ is an empirical covariance matrix of the collection of vectors $\mathcal{F}(\mathcal{S})$. It captures the *second-order* empirical statistical properties of the collection. Note, that the volume of a silhouette tunnel $|\mathcal{S}|$ is typically more than 10^4, often more than 10^5 (e.g., a $128 \times 128 \times 20$ video). Since a covariance matrix is symmetric, only $(13^2+13)/2 = 91$ of its entries are independent thus affording a low-dimensional representation of all feature samples, independently of their number.

4.1.3 Normalization for spatial scale invariance

The shape covariance matrix $C_{\mathcal{S}}$ in (4.2) computed from the 13 features in (4.1) is not invariant to *spatial* scaling of the silhouette tunnel, i.e., two silhouette tunnels $\mathcal{S}$ and $\mathcal{S}'$ that have identical shape but differ in spatial scale will have different covariance matrices. To illustrate the problem, ignoring integer-valued constraints, let $a > 0$ be a spatial scale factor and let $\mathcal{S}' := \{(ax, ay, t)^T : (x, y, t)^T \in \mathcal{S}\}$ be a silhouette tunnel obtained from $\mathcal{S}$ by stretching the horizontal and vertical dimension (but not time) by the factor a. Then $|\mathcal{S}'| = a^2|\mathcal{S}|$. Consider the covariance between the x-coordinate and the distance to the top boundary d_N (both are spatial features)

for both $\mathcal{S}$ and $\mathcal{S}'$. These are respectively given by $\text{cov}(X, D_N)$ and $\text{cov}(X', D'_N)$ where $X' = aX$ and $D'_N = aD_N$. Consequently, $\text{cov}(X', D'_N) = a^2\text{cov}(X, D_N)$. An identical relationship holds for the covariance between any pair of spatial features. The covariance between any spatial feature and any temporal feature for $\mathcal{S}'$ will be a times that for $\mathcal{S}$ (instead of a^2) and the covariance between any pair of temporal features for $\mathcal{S}'$ and $\mathcal{S}$ will be equal. To see how the shape covariance matrix can be made invariant to spatial scaling of the silhouette tunnel, observe that $\text{cov}(X'/\sqrt{|\mathcal{S}'|}, D'_N/\sqrt{|\mathcal{S}'|}) = \text{cov}(X/\sqrt{|\mathcal{S}|}, D_N/\sqrt{|\mathcal{S}|})$. Thus, in order to obtain a spatially scale-invariant shape covariance matrix, we must divide every spatial feature by the square root of the volume of the silhouette tunnel before computing the empirical covariance matrix using (4.2).

A similar approach can be used for temporal scaling which can arise due to frame rate differences between the test and dictionary action segments. However, since most cameras run at either 15 or 30 frames/fields per second, in this work we assume that the frame rates are identical and the segment size N is the same for the test and dictionary action segments. By construction, the shape covariance matrix is automatically invariant to spatio-temporal translation of the silhouette tunnel. It is, however, not invariant to rotation of the silhouette tunnel about the horizontal, vertical, and temporal axes. Rotations about the temporal axis by multiples of 45° have the effect of permuting the spatial components of the feature vector. In this work, however, we will assume that the test and dictionary silhouette tunnels have roughly the same spatial orientation. Rotations about the horizontal and vertical axes are less of a problem in practice because they may not correspond to meaningful real-world silhouette tunnels.

4.2 Optical flow features

In Section 4.1 we have introduced 13-dimensional silhouette shape feature vectors. However, silhouette tunnels are sometimes noisy and unreliable due to the complexity of real-life environments and intrinsic deficiencies of background subtraction algorithms. Thus, we explore another type of local feature vectors, obtained from optical flow of action videos. There have been hundreds of papers written in the past few decades focusing on optical flow computation. Here we use a variant of the Horn-Schunck method, which optimizes a functional based on residuals from the intensity constraint and a smoothness regularization term (Zach et al., 2007). Let $I(x, y, t)$ denote the raw video sequence and let $\mathbf{u}(x, y, t)$ represent the corresponding optical flow vector $\mathbf{u} = (u, v)$ at each pixel position (x, y, t). Based on $I(x, y, t)$ and $\mathbf{u}(x, y, t)$, we define the following feature vector $\mathbf{f}(x, y, t)$ (Guo et al., 2010c):

$$\mathbf{f}(x, y, t) := [x, y, t, I_t, u, v, u_t, v_t, Div, Vor, Gten, Sten]^T, \tag{4.3}$$

where $(x, y, t)^T \in \mathcal{A}$. I_t is the 1-st order partial derivative of $I(x, y, t)$ with respect to t, i.e.,

$$I_t = \frac{\partial I(x, y, t)}{\partial t},$$

u and v are optical flow components, and u_t and v_t are respectively the 1-st order partial derivatives of $u(x, y, t)$ and $v(x, y, t)$ with respect to t. Div is the spatial divergence of the flow field and is defined at each pixel position as:

$$Div(x, y, t) = \frac{\partial u(x, y, t)}{\partial x} + \frac{\partial v(x, y, t)}{\partial y}. \tag{4.4}$$

Divergence can capture the amount of local expansion in the fluid, which can indicate action differences. Vor is the vorticity of flow fields and is defined as:

$$Vor(x, y, t) = \frac{\partial v(x, y, t)}{\partial x} - \frac{\partial u(x, y, t)}{\partial y}. \tag{4.5}$$

In fluid dynamics, vorticity is used to measure local spin around the axis perpendicular to the plane of the flow field. Thus, it is useful to highlight locally circular motion of the moving object. Before explaining $Gten$ and $Sten$, we need to introduce two matrices, called gradient tensor of optical flow $\nabla\mathbf{u}(x, y, t)$ and rate of strain tensor $S(x, y, t)$:

$$\nabla\mathbf{u}(x, y, t) = \begin{pmatrix} \frac{\partial u(x,y,t)}{\partial x} & \frac{\partial u(x,y,t)}{\partial y} \\ \frac{\partial v(x,y,t)}{\partial x} & \frac{\partial v(x,y,t)}{\partial y}, \end{pmatrix} \tag{4.6}$$

$$S(x, y, t) = \frac{1}{2}(\nabla\mathbf{u}(x, y, t) + \nabla\mathbf{u}(x, y, t)^T). \tag{4.7}$$

$Gten$ and $Sten$ are tensor invariants that remain constant no matter what coordinate system they are referenced in. They can be written as follows:

$$Gten(x, y, t) = \frac{1}{2}(tr(\nabla\mathbf{u}(x, y, t))^2 - tr(\nabla\mathbf{u}(x, y, t)^2)), \tag{4.8}$$

$$Sten(x, y, t) = \frac{1}{2}(tr(S(x, y, t))^2 - tr(S(x, y, t)^2)), \tag{4.9}$$

where $tr(\cdot)$ stands for the trace operation.

Similarly to silhouette-based action representation, we compute the empirical 12×12 covariance matrix C_S of feature vectors. Note that only some of these feature vectors are related to the action of moving objects while the remaining feature vectors just indicate background characteristics. Therefore, we select feature vectors associated with a moving object whose corresponding I_t is greater than a threshold.

Only those feature vectors whose corresponding I_t is greater than some threshold are used to calculate the covariance matrix. Fig. 4·5 shows a block diagram of the action representation based on optical flow feature vectors.

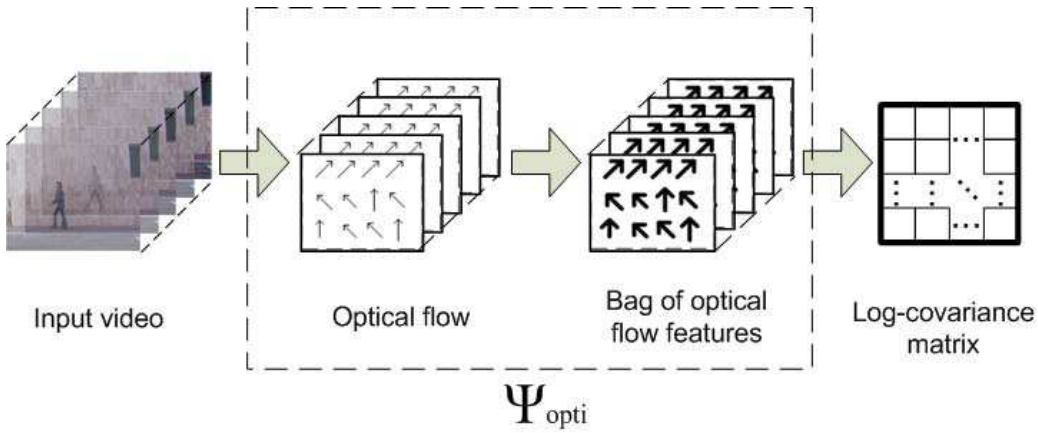

Figure 4·5: Operator Ψ_{opti} that extracts 12×12 covariance matrix of an action in the video.

Chapter 5

Practical considerations

We have described our proposed framework for action recognition, as well as two examples of localized features. There are still some practical considerations in action recognition that we need to consider (Guo et al., 2010b).

5.1 Processing continuous video

In practical cases, we usually need to recognize actions in a continuous video. However, with limited memory, sometimes it is impossible to process the whole video. It is therefore necessary to partition a video into segments (Fig. 5·1), and each time we just need to process a video segment. Additionally, using video segments can enhance the robustness of our action recognition approach. A query video, without being partitioned into segments, can only lead to one feature log-covariance matrix. If it is misclassified, there will be no chance to obtain the correct label for the query action. On the contrary, if the query video is divided into segments, it can still be correctly classified even if a few segments are misclassified.

5.2 Length of segments

Many actions, such as walking, running, jumping etc., are roughly repetitive (Fig. 5·2). Therefore, for repetitive actions, an appropriate segment length is the approximate number of frames in an action period. The typical period for many human actions is on the order of 0.4-0.8 second (except for very fast or very slow motion). For a

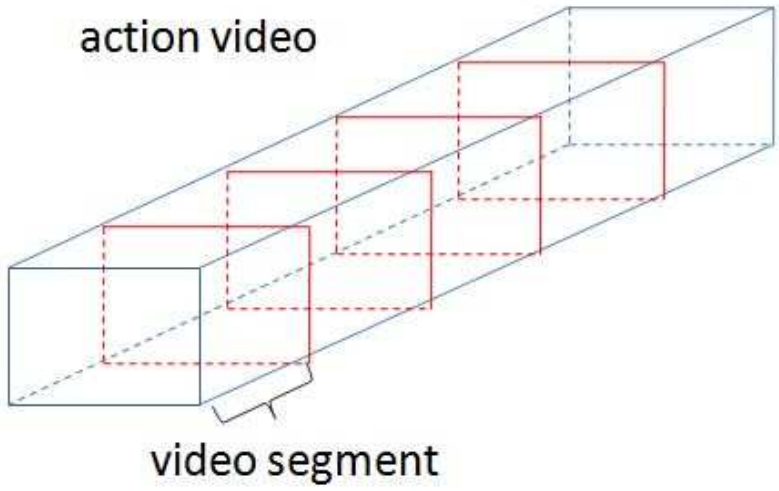

Figure 5·1: Illustration of video segments.

camera operating at 25 fps, the typical length of an action segment is 10–20 frames.

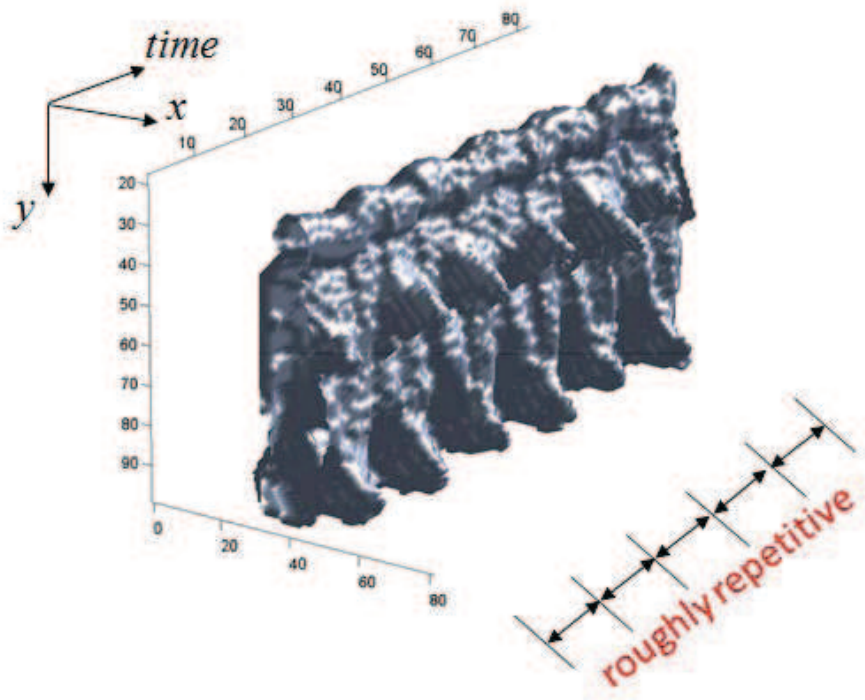

Figure 5·2: Illustration of the repetitive nature of action "walking"

5.3 Temporal misalignment

Suppose there are two videos A and B that include the same action class. Both of them are partitioned into video segments. It is possible that a video segment from A cannot find a good match in video B, due to temporal misalignment. This motivates the need to break a video sequence into successive *overlapping* action segments, as shown in Fig. 5·3. By doing so, actions in video segments can be better synchronized and it is more likely to find well-matched segments. Overlapping video segments can also enrich the training set so that action classification can be more reliable.

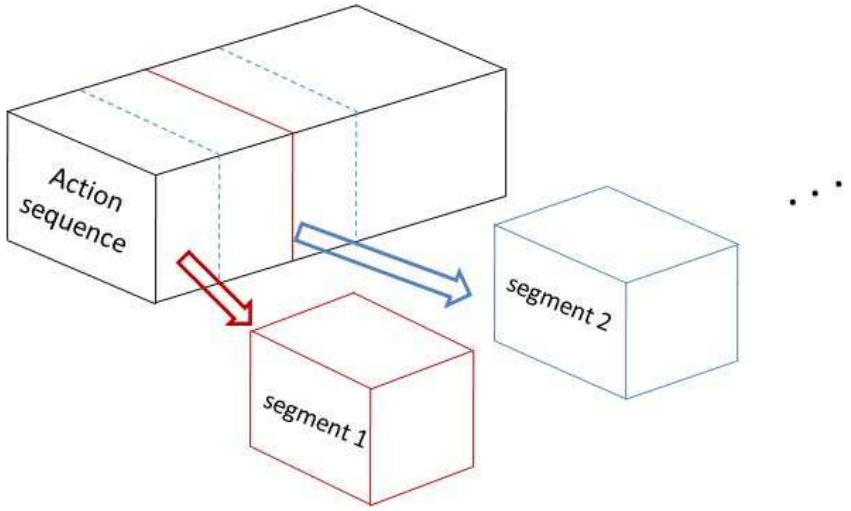

Figure 5·3: Illustration of overlapping segments.

5.4 Majority rule

After partitioning a query video into overlapping segments, we can apply our action recognition approach to each video segment and obtain a sequence of action labels. Suppose each query video only involves one action class, then segment-level action labels can be fused into sequence-level decision based on the majority rule, which take the most frequent segment label as the label of sequence. An example is shown in Fig. 5·4.

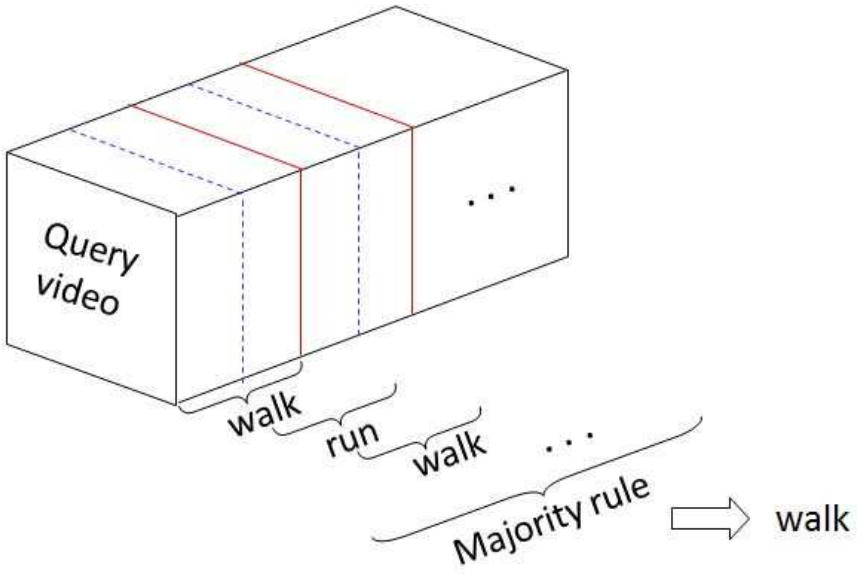

Figure 5·4: Illustration of the majority rule.

5.5 Summary of overall approach

Based on our action recognition framework and practical considerations, the overall approach can be summarized as follows:

1. Partition videos into overlapping action segments (Fig. 5·5);
2. Represent each action segment using the log-covariance matrix of features (Fig. 5·6);
3. Classify each query segment based on NN or SLA classification algorithms (Fig. 5·7);
4. Use the majority rule to decide the action label of the query video (Fig. 5·4).

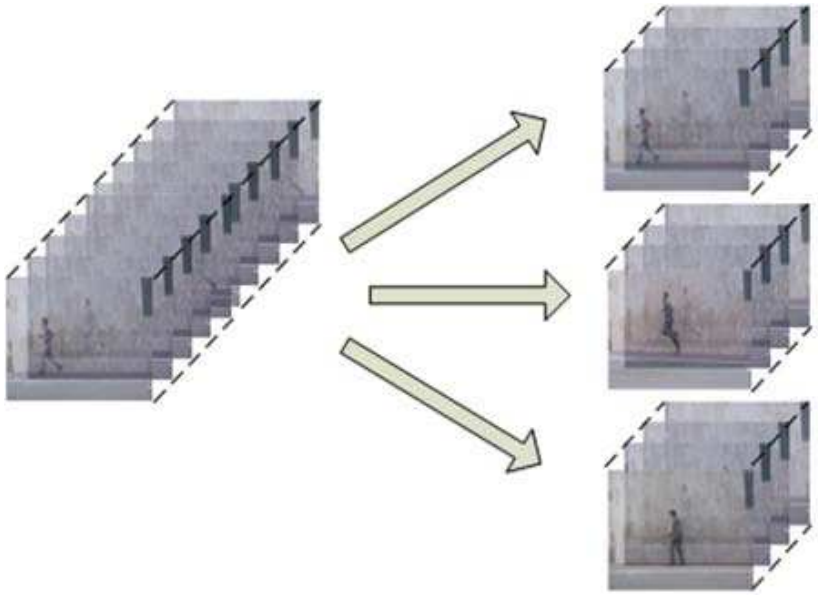

Figure 5·5: Illustration of segment partitioning.

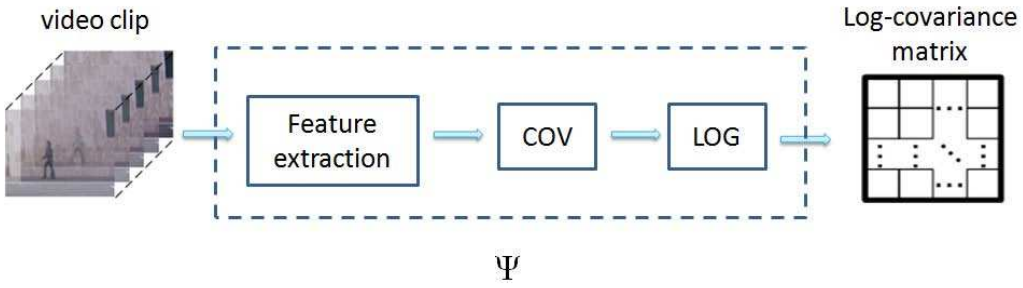

Figure 5·6: Illustration of action representation.

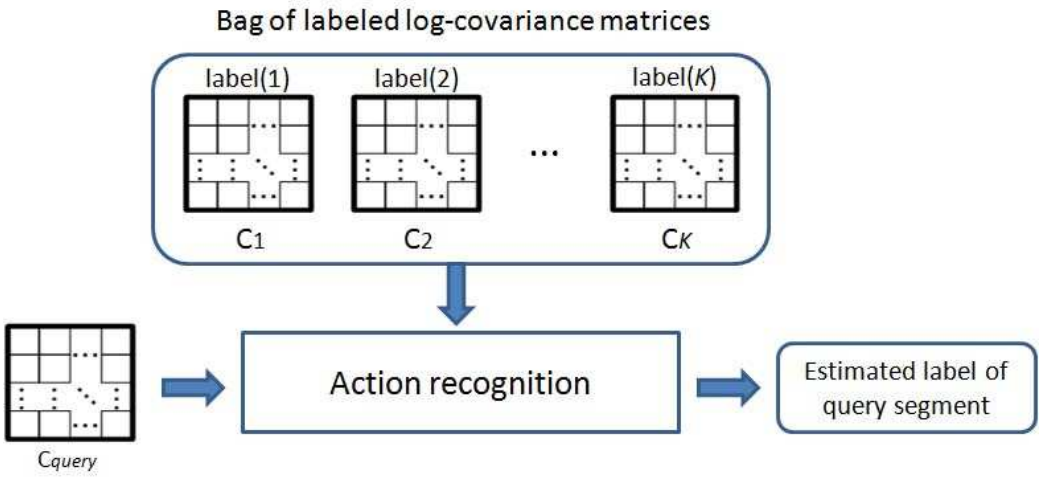

Figure 5·7: Illustration of action classification.

Chapter 6

Experimental results

In previous chapters, we introduced the overall approach for action representation. Specifically, we used two different features based either on a silhouette tunnel or optical flow to represent action dynamics. Then, NN and SLA classifiers were respectively employed for action classification. Thus, there are four possible implementations (2 types of feature vectors $\times$ 2 types of classifiers) to recognize actions. We evaluate their performance on four publicly-available datasets: Weizmann (Gorelick et al., 2007), KTH (Schuldt et al., 2004), UT-Tower (Chen et al., 2010) and Youtube action datasets (Liu et al., 2009).

Our performance evaluation was based on leave-one-out cross validation (LOOCV), as illustrated in Fig. 6·1. First, we divided each video sequence into N-frame long overlapping action segments. Then, we selected one of the action segments as a query segment and used the remaining segments as the training set (except those segments that came from the same video sequence as the query segment). Finally, we identified action class of the query segment. We repeated the procedure for all query segments in the dataset and calculated the correct classification rate (CCR) as the percentage of query segments that were correctly classified. We call this rate the segment-level CCR, or SEG-CCR. In practice, however, one is usually interested in classification of a complete video sequence instead of one of its segments. Since segments provide time-localized action information, in order to obtain classification for the complete video sequence we employed the majority rule (dominant label wins) to all segments

in this sequence. This produces a sequence-level CCR, or SEQ-CCR, defined as the percentage of query sequences that are correctly classified.

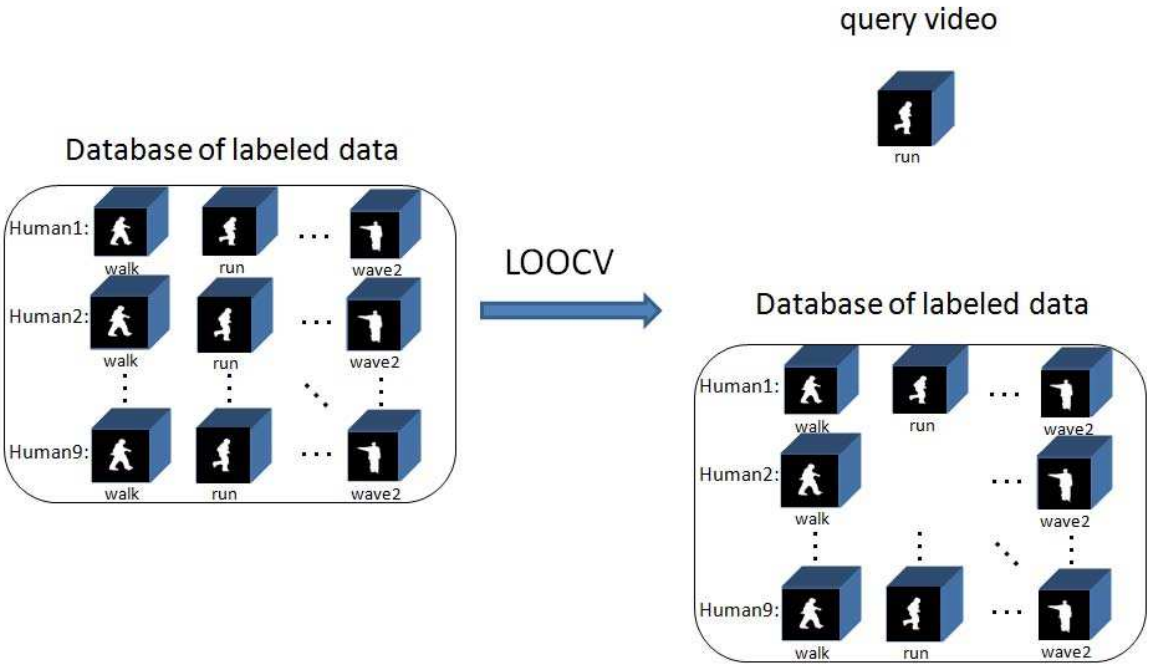

Figure 6·1: Illustration of leave-one-out cross validation.

We also tested our method using "leave-part-out" cross validation (LPOCV), a more challenging test sometimes reported in the literature (Fig. 6·2). In LPOCV, we divided the action segments into non-overlapping training set and test set. After selecting a test segment from the test set, we assigned a class label based on the training set. We repeated this procedure for all test segments. The main difference between LPOCV and LOOCV is that the training set in LPOCV is fixed with less training samples than in LOOCV if both are based on the same dataset. Thus, it is expected that LPOCV will attain poorer performance than LOOCV if other settings remain the same.

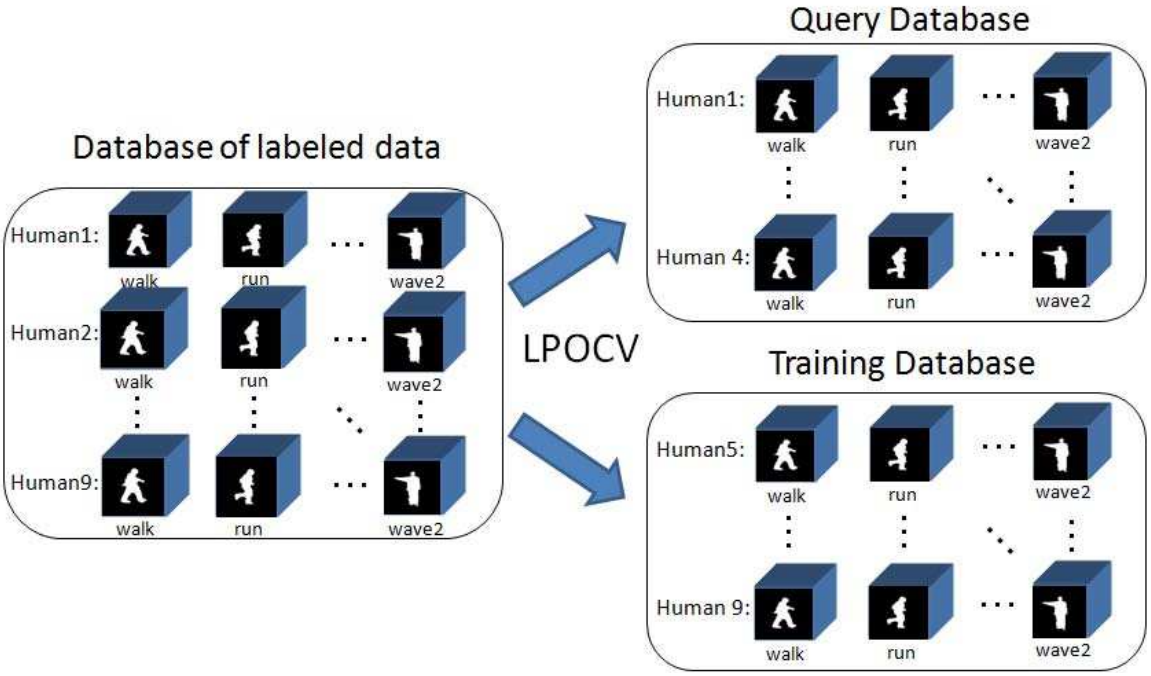

Figure 6·2: Illustration of leave-part-out cross validation.

6.1 Tests on the Weizmann dataset

We conducted a series of experiments on the Weizmann Human Action Database available online[1] (Gorelick et al., 2007). Although this is not a very challenging dataset, many state-of-the-art approaches report performance on it thus affording an easy comparison. The database contains 90 low-resolution video and silhouette sequences (180×144 pixels) that show 9 different people each performing 10 different actions, such as jumping, walking, running, skipping, etc. Some action examples are shown in Fig. 6·3.

We first measure the performance of our action recognition method using silhouette features, and then using optical flow features.

[1] http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html

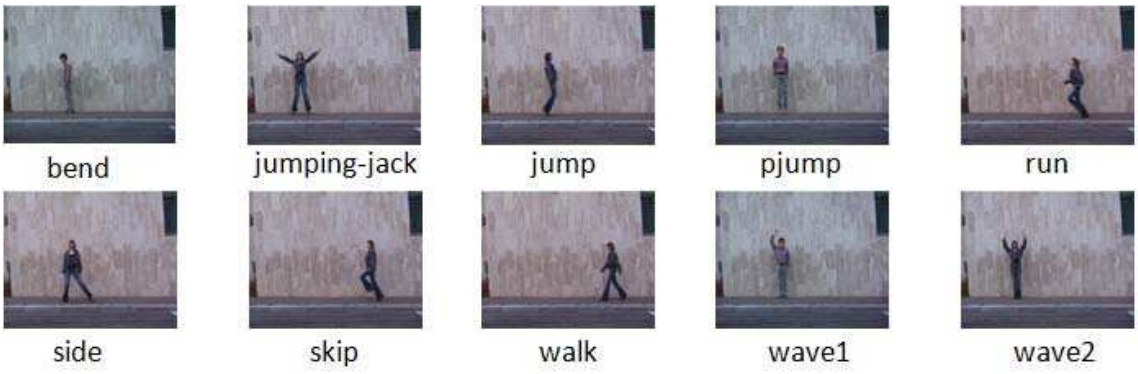

Figure 6·3: Action examples from Weizmann dataset.

6.1.1 Silhouette features

We tested the action recognition performance using NN and SLA classifiers. When we use NN classifier, we obtain SEG-CCR of 97.05% and SEQ-CCR of 100% for segment length $N = 8$. Table 6.1 shows the action "confusion" matrix based on SEG-CCR. The element in row i and column j of the matrix indicates the percentage of action i segments which were classified as action j. The sum of all elements in every row is 100%. The confusion matrix indicates that while some actions are more confusing, such as "skipping" and "jumping", others are easier to distinguish, such as "bending" and "jumping-jack".

If we use the SLA classifier, we obtain SEG-CCR of 96.74% and SEQ-CCR of 100% for $N = 8$. Table 6.2 shows the segment-based action "confusion" matrix. In order to compare the efficacy of NN and SLA classifier, we compute their SEG-CCR and SEQ-CCR under different segment length ($N = 8$ and $N = 20$) and different cross validation methods (LOOCV and LPOCV), as shown in Table 6.3. In Table 6.4, we compare the proposed method with some recent action recognition algorithms that are based on LOOCV.

The above experimental results indicate that:

Table 6.1: Segment-based action confusion matrix for the proposed method (Silhouette features + NN classifier) on 8-frame segments (SEG-CCR = 97.05%).

	bend	jack	jump	sjump	run	side	skip	walk	wave1	wave2
bend	98.6	0	0	0	0	0	0	0	0	1.4
jack	0	100	0	0	0	0	0	0	0	0
jump	0	0	96.1	0	0	0	3.9	0	0	0
sjump	0	0	0	99.3	0	0	0	0	0	0.7
run	0	0	0	0	94.0	1.2	2.4	2.4	0	0
side	0	0	0	0	0	100	0	0	0	0
skip	0	0	1.0	0	10.3	0	86.7	2.1	0	0
walk	0	0	0	0	1.3	0	0	98.7	0	0
wave1	0	0	0	0	0	0	0	0	98.0	2.0
wave2	0	0	0	0	0	0	0	0	4.9	95.1

Table 6.2: Segment-based action confusion matrix for the proposed method (optical flow features + SLA classifier) on 8-frame segments (SEG-CCR = 96.74%).

	bend	jack	jump	sjump	run	side	skip	walk	wave1	wave2
bend	91.9	1.3	0	0.7	0	0	0	0	4.1	2.0
jack	0	99.4	0	0.6	0	0	0	0	0	0
jump	0		95.1	0	0	2.0	2.9	0	0	0
sjump	0	0.8	0	96.7	0	2.5	0	0	0	0.7
run	0	0	0	1.2	91.6	0	1.2	6.2	0	0
side	0	0	0	0	0	100	0	0	0	0
skip	0	0	1.0	0	4.2	0	92.7	2.1	0	0
walk	0	0	0	0	0	0	0	100	0	0
wave1	0	0	0	0.6	0	0	0	0	99.4	0
wave2	0	1.4	0	0	0	0	0	0	1.4	97.2

Table 6.3: Comparison of NN and SLA classifiers based on silhouette features for Weizmann dataset.

		NN classifier		SLA classifier	
		SEG-CCR	SEQ-CCR	SEG-CCR	SEQ-CCR
N=8	LOOCV	97.05%	100%	96.74%	100%
	LPOCV	90.88%	91.11%	91.35%	95.56%
N=20	LOOCV	98.68%	100%	99.49%	100%
	LPOCV	91.82%	95.56%	93.61%	95.56%

- the proposed framework for action recognition achieves remarkable performance (up to 97% SEG-CCR and 100% SEQ-CCR);

Table 6.4: Comparison of silhouette-based action recognition (N=8) using LOOCV on Weizmann dataset.

Method	**NN classifier**	**SR-based classifier**	Gorelick	Niebles	Ali	Seo	Xie	Ikizler
SEG-CCR	**97.05%**	**96.74%**	97.83%	-	95.75%	-	-	-
SEQ-CCR	**100%**	**100%**	-	90%	-	96%	95.6%	100%

- larger N leads to better classification performance;
- using LPOCV results in lower CCR than using LOOCV since LPOCV has a smaller training set and accordingly is more challenging;
- in most cases, the SLA classifier has similar or better performance compared to the NN classifier;
- the majority rule improves the recognition performance.

6.1.2 Optical flow features

In contrast to previous tests, we now use optical flow features to represent action dynamics, and either NN or SLA classifier. For the NN classifier, we obtain SEG-CCR of 89.74% and SEQ-CCR of 91.11% ($N = 8$). Table 6.5 shows the action confusion matrix based on SEG-CCR. If we use the SLA classifier, we attain SEG-CCR of 92.69% and SEQ-CCR of 94.44%. Table 6.6 shows the action "confusion" matrix based on SEG-CCR. Additionally, we compute their SEG-CCR and SEQ-CCR under different segment length ($N = 8$ and $N = 20$) and different cross validation methods (LOOCV and LPOCV), as shown in Table 6.7.

6.1.3 Variation of CCR in LPOCV

In LPOCV we partition an action dataset into non-overlapping query set and training set, and then recognize actions in query videos. However, our previous results only show one particular leave-part-out partition, which may not be indicative of the

Table 6.5: Segment-based action confusion matrix for the proposed method (optical flow features + NN classifier) on 8-frame segments (SEG-CCR = 89.74%).

	bend	jack	jump	sjump	run	side	skip	walk	wave1	wave2
bend	100	0	0	0	0	0	0	0	0	0
jack	0	99.4	0	0.6	0	0	0	0	0	0
jump	0	0	66.7	0	7.1	6.1	20.1	0	0	0
sjump	0	5.8	0	93.3	0	0.8	0	0	0	0
run	0	0	1.3	0	69.6	0	27.9	1.3	0	0
side	0	0	4.2	0	3.2	83.2	3.2	6.3	0	0
skip	0	0	11.7	0	29.8	0	58.5	0	0	0
walk	0	0	0	0	0	0	0	100	0	0
wave1	0	0	0	0	0	0	0	0	98.7	1.3
wave2	0	0	0	0	0	0	0	0	1.4	98.6

Table 6.6: Segment-based action confusion matrix for the proposed method (optical flow features + SLA classifier) on 8-frame segments (SEG-CCR = 92.69%).

	bend	jack	jump	sjump	run	side	skip	walk	wave1	wave2
bend	98.4	0	0	0	0	0	0	0	1.6	0
jack	0	100	0	0	0	0	0	0	0	0
jump	0	0	71.7	0	1.0	4.0	19.2	3.0	0	1.0
sjump	0	0.9	0	99.1	0	0	0	0	0	0.7
run	0	0	1.3	0	82.3	1.3	8.9	5.0	1.2	0
side	0	0	2.1	0	0	96.9	0	1.0	0	0
skip	0	0	11.2	0	12.8	0	74.0	2.0	0	0
walk	0	0	0	0	0	0	0	100	0	0
wave1	3.3	0	0	0	0	0	0	0.7	94.0	2.0
wave2	0	0	0	0	0	0	0	0	0	100

Table 6.7: Comparison of NN and SLA classifiers based on optical flow features for Weizmann dataset.

		NN classifier		SR-based classifier	
		SEG-CCR	SEQ-CCR	SEG-CCR	SEQ-CCR
N=8	LOOCV	89.74%	91.11%	92.69%	94.44%
	LPOCV	79.45%	80.00%	83.20%	88.89%
N=20	LOOCV	91.93%	92.22%	94.09%	94.44%
	LPOCV	81.80%	82.22%	87.35%	88.89%

overall classification performance under LPOCV. Thus, we randomly partition the action dataset in 100 trails, and each time we compute SEG-CCRs and SEQ-CCRs

under each partition. Finally, we can compute the mean and variance of all SEG-CCRs and SEQ-CCRs, to obtain an average classification performance and its spread, as shown in Table 6.8.

Table 6.8: Mean and standard deviation (Std) of SEG-CCRs and SEQ-CCRs based on random partitions in LPOCV using NN classifier.

Feature	Mean SEG-CCR	Std SEG-CCR	Mean SEQ-CCR	Std SEQ-CCR
Silhouette	94.95%	1.84%	98.56%	2.12%
Optical flow	82.45%	4.12%	84.56%	5.10%

We see that the mean of SEG-CCR and SEQ-CCR in Table 6.8 are greater than the corresponding CCRs in Tables 6.3 and 6.7. It implies that the particular leave-part-out partition in Tables 6.3 and 6.7 was more challenging than the 100 partitions on average. Additionally, it is clear that the variances of SEG-CCR and SEQ-CCR are small. Therefore, we see that our approach is robust to different partitions in LPOCV.

6.2 Tests on the KTH dataset

We then conducted experiments on the KTH action dataset. This dataset contains six different human actions: handclapping, handwaving, walking, jogging, running and boxing, performed repeatedly by 25 people in 4 different scenarios (outdoor, outdoor with zoomed camera, outdoor with different clothes, and indoor). Action examples are shown in Fig. 6·4. This is a more challenging dataset because:

- the camera is no longer static (vibration and zoom-in/zoom-out);
- there are large variations in human body shape, view angles, scales and appearance.

Due to the difficulties mentioned above, we are not able to obtain a reliable silhouette shape representation of actions.

Figure 6·4: Action examples from KTH dataset.

Thus, in this section, we test our methods using optical flow features, combined with NN classifier and SLA classifier. For the NN classifier and LOOCV, the SEG-CCR is 89.55% and SEQ-CCR is 98.17% ($N = 20$). Tables 6.9 and 6.10 show action confusion matrices based on SEG-CCR and SEQ-CCR for LOOCV with $N = 20$. If we use the SLA classifier and LOOCV with $N = 20$, the SEG-CCR is 90.84% and SEQ-CCR is 98.50%. Likewise, Tables 6.11 and 6.12 show the corresponding action confusion matrices.

Table 6.9: Segment-level action confusion matrix for the proposed method (optical flow features + NN classifier) on 20-frame segments (SEG-CCR = 89.55%).

	clap	wave	walk	jog	run	box
clap	90.6	5.9	2.8	0.1	0.2	0.3
wave	4.9	90.3	4.2	0	0.1	0.5
walk	1.9	3.0	94.9	0.1	1.0	0.1
jog	1.1	0.7	0.4	76.6	13.1	8.1
run	0.3	0.4	0.3	10.3	85.1	3.6
box	0.6	1.7	0.5	3.1	1.3	93.0

In LPOCV test we followed the training/query set breakup as proposed by Schuldt *et al.* (Schuldt et al., 2004). When we use the NN classifier, we obtain SEG-CCR of 85.42% and SEQ-CCR of 96.88%. When we use the SLA classifier, we obtain SEG-CCR of 86.04% and SEQ-CCR of 97.40% in this case.

Table 6.10: Sequence-level action confusion matrix for the proposed method (optical flow features + NN classifier) on 20-frame segments (SEQ-CCR = 98.17%).

	clap	wave	walk	jog	run	box
clap	100	0	0	0	0	0
wave	0	100	0	0	0	0
walk	0	1	99	0	0	0
jog	0	0	0	97	2	1
run	0	0	0	6	93	1
box	0	0	0	0	0	100

Table 6.11: Segment-based action confusion matrix for the proposed method (optical flow features + SLA classifier) on 20-frame segments (SEG-CCR = 90.84%).

	clap	wave	walk	jog	run	box
clap	94.1	2.8	0.8	0.1	0.2	2.0
wave	4.5	90.9	2.7	0.4	0.4	1.2
walk	1.2	1.2	97.4	0	0.1	0.2
jog	0.2	0.5	0.2	81.4	10.0	7.7
run	0	0.1	0	10.5	86.4	3.0
box	0.3	0.2	0	4.1	3.1	92.3

Table 6.12: Sequence-level action confusion matrix for the proposed method (optical flow features + SLA classifier) on 20-frame segments (SEQ-CCR = 98.50%).

	clap	wave	walk	jog	run	box
clap	99	1	0	0	0	0
wave	0	100	0	0	0	0
walk	0	0	100	0	0	0
jog	0	0	0	97	2	1
run	0	0	0	5	95	0
box	0	0	0	0	0	100

We compared the proposed method with some recent action recognition algorithms. Table 6.13 shows results for LOOCV tests and Table 6.14 shows results for LPOCV tests. Since we know that LPOCV is more challenging than LOOCV, it is unfair to compare method A that is tested under LOOCV with method B that is

tested under LPOCV. Although the SEG-CCR for our method is a little worse than that for Ali *et al.*'s method (Ali and Shah, 2010), our SEQ-CCRs are in line with the best methods today.

Table 6.13: Comparison of our optical-flow-based approach (N=20) using LOOCV on KTH dataset.

Method	**NN classifier**	**SR-based classifier**	Kim	Wu	Wong	Dollar	Seo
SEG-CCR	**89.55%**	**90.84%**	-	-	81.0%	81.2%	-
SEQ-CCR	**98.17%**	**98.50%**	95.3%	94.5%	-	-	95.7%

Table 6.14: Comparison of the optical-flow-based approach (N=20) using LPOCV on KTH dataset.

Method	**NN classifier**	**SR-based classifier**	Ali	Laptev	Le	Wang	Kovashka
SEG-CCR	**85.42%**	**86.04%**	87.7%	-	-	-	-
SEQ-CCR	**96.88%**	**97.40%**	-	91.8%	93.9%	94.2%	94.5%

6.3 Tests on the UT-Tower dataset

We also conducted experiments on the UT-Tower action dataset. The UT-tower action dataset comes from the "Aerial View Activity Classification Challenge" in ICPR 2010 Contest on Semantic Description of Human Activities (SDHA). This dataset contains video sequences of a single person performing various actions recorded from the top of the University of Texas at Austin's main tower. The dataset consists of 108 video sequences of 360 $\times$ 240 pixel resolution and frame rate of 10 fps. The contest required classifying video sequences into 9 categories of human actions: {1: pointing, 2: standing, 3: digging, 4: walking, 5: carrying, 6: running, 7: wave1, 8: wave2, 9: jumping}. Each of the 9 actions is performed 2 times by 6 individuals for a total of 12 video sequences per action category. The pointing, standing, digging, and walking video sequences have a concrete background whereas the carrying, running, wave1, wave2, and jumping video sequences have a lawn background. The cameras

are stationary but have jitter. The average height of human figures in this dataset is about 20 pixels. In addition to the challenges associated with the low resolution of objects of interest in this dataset, there are additional challenges from shadows and blurry visual cues. Ground truth action labels for all video sequences were provided for training and testing. In addition, in order to alleviate segmentation and tracking issues and make participants focus on the classification problem, ground truth bounding boxes as well as foreground masks for each video sequence were also provided. Action samples are shown in Fig. 6·5.

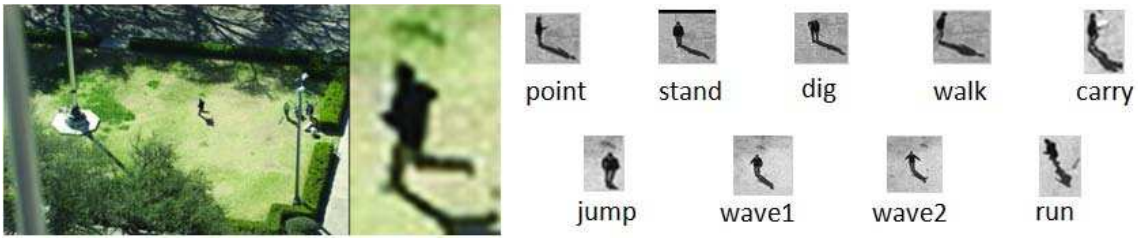

Figure 6·5: Action examples from UT-Tower dataset.

6.3.1 Silhouette features

We first tested action recognition performance using the NN classifier and we obtained SEG-CCR of 93.53% and SEQ-CCR of 96.30% ($N = 8$). Tables 6.15 and 6.16 show the action "confusion" matrices based on SEG-CCR and SEQ-CCR. For the SLA classifier, we obtained SEG-CCR of 96.15% and SEQ-CCQ of 97.22%. Tables 6.17 and 6.18 show the action "confusion" matrix based on SEG-CCR and SEQ-CCR.

6.3.2 Optical flow features

Then, we used optical flow features to represent action dynamics, together with NN and SLA classifiers. For the NN classifier, we obtained SEG-CCR of 82.25% and SEQ-CCR of 86.11% ($N = 8$). Tables 6.19 and 6.20 show the corresponding action "confusion" matrices based on SEG-CCR and SEQ-CCR. For the SLA classifier, we

Table 6.15: Segment-based action confusion matrix for the proposed method (silhouette features + NN classifier) on 8-frame segments (SEG-CCR = 93.53%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	72.3	18.0	9.7	0	0	0	0	0	0
stand	5.8	92.8	1.4	0	0	0	0	0	0
dig	4.5	0.5	94.5	0	0	0	0.5	0	0
walk	0	0	0	97.3	0	2.7	0	0	0
carry	0	0	0	0	97.7	2.3	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0	0	0	0	85.6	14.4	0
wave2	0	0	0	0	0	0	0	100	0
jump	0	0	0	0	0	1.0	0	0	99.0

Table 6.16: Sequence-based action confusion matrix for the proposed method (silhouette features + NN classifier) on 8-frame segments (SEQ-CCR = 96.30%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	75.0	16.7	8.3	0	0	0	0	0	0
stand	8.3	91.7	0	0	0	0	0	0	0
dig	0	0	100	0	0	0	0	0	0
walk	0	0	0	100	0	0	0	0	0
carry	0	0	0	0	100	0	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0	0	0	0	100	0	0
wave2	0	0	0	0	0	0	0	100	0
jump	0	0	0	0	0	0	0	0	100

Table 6.17: Segment-based action confusion matrix for the proposed method (silhouette features + SLA classifier) on 8-frame segments (SEG-CCR = 96.15%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	88.0	6.0	6.0	0	0	0	0	0	0
stand	4.4	94.2	1.4	0	0	0	0	0	0
dig	2.0	1.5	96.0	0	0.5	0	0	0	0
walk	1.4	0	0	98.6	0	0	0	0	0
carry	0	0	0	0	99.5	0.5	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0.5	0	0	0	94.1	5.4	0
wave2	0	0	0	0	0	0	7.5	92.5	0
jump	0	0	0	0	0	0	0	0	100

Table 6.18: Sequence-based action confusion matrix for the proposed method (silhouette feature vectors + SLA classifier) on 8-frame segments (SEQ-CCR = 97.22%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	91.7	0	8.3	0	0	0	0	0	0
stand	16.7	83.3	0	0	0	0	0	0	0
dig	0	0	100	0	0	0	0	0	0
walk	0	0	0	100	0	0	0	0	0
carry	0	0	0	0	100	0	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0	0	0	0	100	0	0
wave2	0	0	0	0	0	0	0	100	0
jump	0	0	0	0	0	0	0	0	100

obtained SEG-CCR of 81.18% and SEQ-CCR of 85.19%. Tables 6.21 and 6.22 show segment-based and sequence-based action "confusion" matrices.

Table 6.19: Segment-based action confusion matrix for the proposed method (optical flow features + NN classifier) on 8-frame segments (SEG-CCR = 82.25%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	53.0	27.7	14.5	0	0	0	4.8	0	0
stand	45.5	51.5	1.5	0	0	0	1.5	0	0
dig	1.5	1.0	96.5	0	0	0	0.5	0.5	0
walk	0	0	0	93.2	4.1	2.7	0	0	0
carry	0	0	0	0	87.6	12.0	0.5	0	0
run	0	0	0	0	9.1	90.9	0	0	0
wave1	1.1	1.6	0	0	0	0	66.9	30.4	0
wave2	0	0	0	0	0	0	17.5	82.5	0
jump	0	0	0	0	0	0	0	0	100

Clearly, the proposed silhouette-based action recognition outperforms its optical-flow-based counterpart by over 10%. This is not surprising when one closely examines two special actions: pointing and standing. These actions, strictly speaking, are not actions as they involve no movement. Thus, optical flow computed in each case is zero (except for noise and errors) leading to failure of optical-flow-based approaches. On the other hand, silhouette-based approaches are less affected since pointing and

Table 6.20: Sequence-based action confusion matrix for the proposed method (optical flow features + NN classifier) on 8-frame segments (SEQ-CCR = 86.11%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	83.3	8.3	8.3	0	0	0	0	0	0
stand	66.7	25.0	8.3	0	0	0	0	0	0
dig	0	0	100	0	0	0	0	0	0
walk	0	0	0	100	0	0	0	0	0
carry	0	0	0	0	100	0	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0	0	0	0	75.0	25.0	0
wave2	0	0	0	0	0	0	8.3	91.7	0
jump	0	0	0	0	0	0	0	0	100

Table 6.21: Action confusion matrix for the proposed method (optical flow feature + SR-based classifier) on 8-frame segments (SEG-CCR = 81.18%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	53.0	35.0	8.4	0	0	0	3.6	0	0
stand	36.4	55.7	6.82	0	0	0	1.1	0	0
dig	1.0	1.0	96.0	0	0	0.5	1.5	0	0
walk	0	0	1.4	91.8	1.4	4.1	0	0	1.4
carry	0	0	0	0	95.9	4.2	0.5	0	0
run	0	0	0	4.6	18.2	77.3	0	0	0
wave1	1.1	0.5	1.1	0	0	0	81.0	16.3	0
wave2	0	0	0	0	0	0	52.6	47.4	0
jump	0	0	0	0	0	0	0	0	100

Table 6.22: Action confusion matrix for the proposed method (optical flow feature + SR-based classifier) on 8-frame segments (SEQ-CCR = 85.19%).

	point	stand	dig	walk	carry	run	wave1	wave2	jump
point	66.7	33.3	0	0	0	0	0	0	0
stand	33.3	66.7	0	0	0	0	0	0	0
dig	0	0	100	0	0	0	0	0	0
walk	0	0	0	100	0	0	0	0	0
carry	0	0	0	0	100	0	0	0	0
run	0	0	0	0	0	100	0	0	0
wave1	0	0	0	0	0	0	91.7	8.3	0
wave2	0	0	0	0	0	0	58.3	41.7	0
jump	0	0	0	0	0	0	0	0	100

standing can still be described by 3-D silhouette shape. Also, note a much higher silhouette-based CCR for wave1. Examining the confusion matrix (not shown here) we realized that around 50% of wave1 videos are misclassified by our optical-flow-based action recognition as wave2. It indicates that the optical-flow features are also not quite discriminative when representing wave1.

6.4 Tests on the YouTube dataset

YouTube dataset is a very complex dataset based on YouTube videos (Liu et al., 2009). This dataset contains 11 action classes that are grouped into 25 groups: basketball shooting, biking/cycling, diving, golf swinging, horse-back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog (Fig. 6·6). This dataset is very challenging due to large variations in camera motion, acquisition viewpoint, cluttered background, etc. Since silhouette tunnels are not available for this dataset, we only tested the optical flow features. For the NN classifier, we obtained SEG-CCR of 50.4% and SEQ-CCR of 78.5% ($N = 20$, LOOCV). Table 6.23 shows the SEQ-CCR comparison of our proposed method with state-of-the-art methods. The performance of our method is in line with state-of-the-art methods today.

Figure 6·6: Action examples from YouTube action dataset.

Table 6.23: Comparison of our proposed approach (N=20) and state-of-art methods based on LOOCV on YouTube dataset.

Method	**Proposed**	Liu	Ikizler	Le	Wang	Noguchi
SEG-CCR	**50.4%**	-	-	-	-	-
SEQ-CCR	**78.5%**	71.2	75.2	75.8%	84.2%	80.4%

6.5 Representation and metric comparison

The experiments so far indicate that feature covariance matrices are sufficiently discriminative for action recognition, and the Euclidean distance based on log-covariance matrices is an appropriate metric. However, how would other representations or metrics fair against them? To answer this question, we performed several experiments using silhouette features and the NN classifier on Weizmann dataset. First, rather than using second-order statistics to characterize localized features we have tested first-order statistics, i.e., the mean, under the Euclidean distance metric. As is clear from Table 6.24, recognition performance using mean representation is vastly inferior to that of covariance representation with log-covariance metric (over 50% drop). Secondly, we used a covariance matrix instead of a log-covariance matrix. Again, the performance dropped dramatically compared to the log-covariance representation. Finally, we assumed that feature vectors are drawn from a Gaussian distribution and we estimated this distribution's mean vector and covariance matrix. Then, we used KL-divergence to measure the distance between two Gaussian distributions. This approach fared much better however it still trailed the log-covariance representation under the Euclidean metric by over 6%.

Table 6.24: Recognition performance for various formulations using silhouette features and NN classifier with N=8 on Weizmann dataset.

Representation Metric	**Log-Covariance Euclidean**	Mean Euclidean	Covariance Euclidean	Gaussian fit KL-divergence
SEG-CCR	**97.1**	45.8	43.6	91.3
SEQ-CCR	**100**	48.9	56.7	93.4

6.5.1 Analysis of feature importance

In Chapter 4, we introduced two examples of local feature vectors: silhouette features and optical flow features, consisting of 13 and 12 features, respectively. It is, therefore, necessary to investigate the contribution of each feature component to the classification performance. In this section, we will analyze the importance of feature components by conducting experiments on the Weizmann dataset based on LOOCV and segment length $N = 8$ using only subsets of feature components.

Silhouette features

The silhouette feature vector $\mathbf{f}(x, y, t)$ is defined in (4.1). In order to verify usefulness of individual features, we partition $\mathbf{f}(x, y, t)$ into three subsets:

- (x, y, t): spatio-temporal coordinates,
- $(d_E, d_W, d_N, d_S, d_{NE}, d_{SW}, d_{SE}, d_{NW})$: spatial distances,
- (d_{T+}, d_{T-}): temporal distances.

The spatio-temporal coordinates provide localization information for moving objects. Spatial and temporal distances describe local spatial and temporal shape deformations, repectively. Table 6.25 shows the classification performance (SEG-CCR and SEQ-CCR) using subsets of feature components on the Weizmann dataset. The results indicate that the subset $(d_{NE}, d_{SW}, d_{SE}, d_{NW})$ is the most significant one. In contrast, the subset (d_{T+}, d_{T-}) contributes the least to action classification. However, the combination of (x, y, t) and $(d_{T+})/(d_{T-})$ leads to a remarkable classification performance (up to 95.56%). Thus, even if (d_{T+}, d_{T-}) themselves are not discriminative enough for action recognition, they can still provide auxiliary action characteristics when combined with other features.

Table 6.25: Classification performances using subsets of silhouette features on Weizmann dataset using LOOCV ($N = 8$).

Subset of feature components	SEG-CCR	SEQ-CCR
(x, y, t)	69.84%	84.44%
(d_E, d_W, d_N, d_S)	69.29%	80.00%
$(d_{NE}, d_{SW}, d_{SE}, d_{NW})$	81.83%	89.99%
(d_{T+}, d_{T-})	33.02%	41.11%
(x, y, t, d_E)	76.59%	90.00%
(x, y, t, d_W)	76.83%	91.11%
(x, y, t, d_N)	82.06%	92.22%
(x, y, t, d_S)	79.60%	90.00%
$(x, y, t, d_E, d_W, d_N, d_S)$	90.71%	96.67%
(x, y, t, d_{SE})	80.56%	94.44%
(x, y, t, d_{NW})	78.65%	88.89%
(x, y, t, d_{SW})	79.68%	92.22%
(x, y, t, d_{NE})	81.35%	93.33%
$(x, y, t, d_{NE}, d_{SW}, d_{SE}, d_{NW})$	91.11%	98.89%
(x, y, t, d_{T+})	84.92%	95.56%
(x, y, t, d_{T-})	85.56%	93.33%
$(x, y, t, d_{T+}, d_{T-})$	88.83%	95.56%
All features	97.05%	100%

Optical-flow features

The optical-flow feature vector is defined in (4.3). The optical-flow feature components can be partitioned into three groups:

- (x, y, t): spatio-temporal coordinates,
- (I_t, u, v, u_t, v_t): optical flow and temporal gradients,
- $(Div, Vor, Gten, Sten)$: optical-flow descriptors derived from fluid dynamics.

Table 6.26 shows the classification performance (SEG-CCR and SEQ-CCR) using subsets of optical-flow feature on the Weizmann dataset. From this table we see that (I_t, u, v, u_t, v_t) is the most significant feature subset for action recognition. Specifically, u is the most discriminative feature when combined with the localization subset (x, y, t), that achieves SEQ-CCR of 87.78%.

Table 6.26: Classification performance using subsets of optical-flow features on Weizmann dataset using LOOCV ($N = 8$).

Selected features	SEG-CCR	SEQ-CCR
(x, y, t)	71.92%	73.33%
(I_t, u, v, u_t, v_t)	72.33%	83.33%
$(Div, Vor, Gten, Sten)$	57.14%	78.89%
(x, y, t, I_t)	73.23%	77.78%
(x, y, t, u)	83.09%	87.78%
(x, y, t, v)	82.43%	84.44%
(x, y, t, u_t)	80.13%	83.33%
(x, y, t, v_t)	79.64%	81.11%
$(x, y, t, I_t, u, v, u_t, v_t)$	85.63%	88.24%
(x, y, t, Div)	79.72%	82.22%
(x, y, t, Vor)	80.54%	81.78%
$(x, y, t, Gten)$	76.35%	77.78%
$(x, y, t, Sten)$	77.59%	81.11%
$(x, y, t, Div, Vor, Gten, Sten)$	81.77%	83.33%
All	89.74%	91.11%

6.5.2 Robustness experiments

The proposed framework has performed well when the query action was similar to dictionary actions. In practice, however, the query action may be "distorted", e.g., person carrying a bag, or may be captured from a different viewpoint. We tested the robustness of our approach to action variability and camera viewpoint on videos originally used by Gorelick (Gorelick et al., 2007) that include 10 walking people in various scenarios (walking with a briefcase, limping, etc.). We tested both silhouette features and optical-flow features using the NN classifier. Experimental results for action variability are shown in Table 6.27. Since there is only one instance of each type of test sequence, SEQ-CCR must be either 100% or 0%. Clearly, all test sequences were correctly labeled even if some segments were misclassified. Also, the optical-flow features perform better overall than the silhouette features (except for "Knees up" at segment level).

Experimental results for viewpoint dependence are shown in Table 6.28. The test videos contain the action of walking captured from different angles as shown in

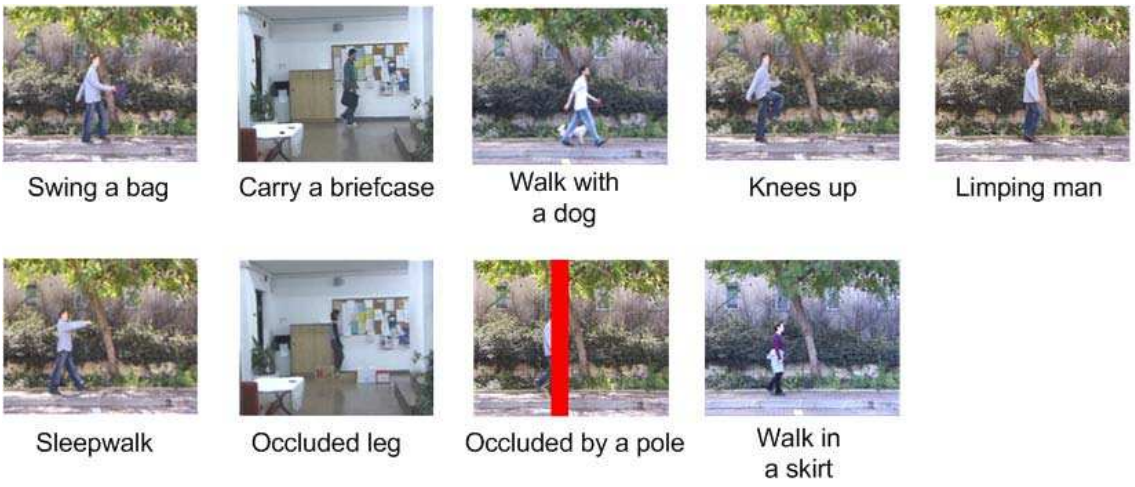

Figure 6·7: Action examples of walking variations.

Table 6.27: Results of a robustness test to action variability; query actions differ significantly from dictionary actions.

	Silhouette features		Optical-flow features	
	SEG-CCR	SEQ-CCR	SEG-CCR	SEQ-CCR
Swing a bag	94.9%	100%	100%	100%
Carry a briefcase	100 %	100%	100%	100%
Walk with a dog	82.4%	100%	100%	100%
Knees up	73.5%	100%	62.7%	100%
Limping man	100 %	100%	100%	100%
Sleepwalk	100 %	100%	100%	100%
Occluded legs	94.9%	100%	100%	100%
Normal walk	100 %	100%	100%	100%
Occluded by a pole	88.1%	100%	100%	100%
Walk in a skirt	100 %	100%	100%	100%

Fig. 6·8 (varying from 0° to 81° with steps of 9° with 0° being the side view). The action samples in the training dataset (Weizmann dataset) are all captured from the side view. Thus, it is expected that the classification performance will degrade when the camera angle increases. The results indicate that silhouette features are robust for walking up to about 36° in viewpoint change and that confusion starts at about 54° (walking recognized as other actions). Optical-flow features perform slightly better in this case; good performance up to about 54° and misclassification starts around 72°.

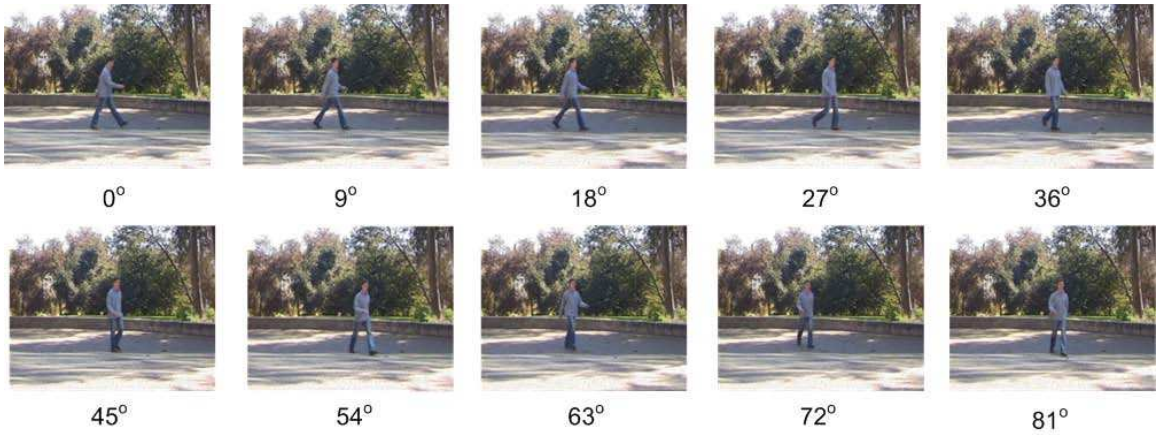

Figure 6·8: Action examples of walking from different viewpoints.

Table 6.28: Results of a robustness test to camera viewpoint; query action captured from different angles than those in the dictionary.

	Silhouette features		Optical-flow features	
	SEG-CCR	SEQ-CCR	SEG-CCR	SEQ-CCR
Viewpoint 0°	100 %	100%	100%	100%
Viewpoint 9°	100 %	100%	100%	100%
Viewpoint 18°	100 %	100%	100%	100%
Viewpoint 27°	92.3%	100%	100%	100%
Viewpoint 36°	90.8%	100%	100%	100%
Viewpoint 45°	76.8%	100%	100%	100%
Viewpoint 54°	38.5%	0%	98.1%	100%
Viewpoint 63°	20.2%	0%	69.1%	100%
Viewpoint 72°	13.8%	0%	40.4%	0%
Viewpoint 81°	5.4 %	0%	30.2%	0%

6.5.3 Computational complexity analysis

The proposed approaches are computationally efficient and easy to implement. Our experimental platform is Intel Centrino (CPU: T7500 2.2GHz + Memory: 2GB) with Matlab 7.6. The extraction of 13-dimensional feature vectors from a silhouette tunnel and calculation of covariance matrices take together about 10.1 sec for a 180×144-pixel, 84-frame silhouette sequence (0.12 sec per frame). The computation of 12-dimensional optical-flow feature vectors and their covariance matrices takes about 6 sec (0.07 sec per frame) for the same sequence. Note that silhouette and optical flow estimation are not included. Given a query sequence with 613 query segments and

a training set with 605 training segments, the NN classifier requires about 52 sec to classify all query segments (0.08 sec per query segment), while the SLA classifier needs about 44 sec (solving 613 times l^1-norm minimization problem, i.e., 0.07 sec per query segment). The computation time indicates that optical-flow features have lower computational complexity than silhouette-based features, and the NN classifier has very close computational complexity to the SLA classifier. This method is also memory efficient, since the training sets and test sets essentially store 13×13 or 12×12 covariance matrices, instead of video data.

Chapter 7

Action change detection

So far we have studied action recognition based on the assumption that each video only contains a single action class. In practical cases, however, a given a video sequence may contain multiple action classes. For instance, a walking pedestrian may stop to hug a friend, and then start running towards an approaching bus. This example involves three actions, and if we parse the whole video into three sub-videos (single action in each sub-video), we can employ our proposed approach to get the three action labels of the whole video. This temporal parsing can be achieved by identifying time instants when an action changes, i.e., detecting when one action stops and another action begins. Finding these temporal action boundaries is akin to scene cut detection in video, but in the space of actions. The problem is fundamental to action analysis; even if actions can be reliably described (i.e., action representation is established) and even if they can be accurately compared (i.e., action comparison metric is known), it is still unclear to what segment of a video should both be applied. In other words, should the action representation be applied to 5, 10, 20 or more frames? Action change detection establishes temporal boundaries between which no action change takes place and thus action representation is meaningful. This is useful when comparing an observed action with a dictionary of annotated actions, but can also be used for unsupervised action segmentation, i.e., identification of actions as A, B, C, ..., that can be later annotated by a human operator.

The detection of abrupt changes is a classical topic that has been widely stud-

ied in the past few decades (Basseville and Nikiforov, 1993). It has been applied in many areas, such as quality control, time series signal analysis, fault detection and monitoring, etc. Many algorithms have been developed for abrupt change detection, such as the cumulative sum algorithm (CUSUM), statistical algorithms, Shewharts control charts, etc. In this thesis, we apply a statistical algorithm to detect abrupt action changes. We develop a sequential, adaptive, and unsupervised action change detection algorithm based on our action representation framework, centered around covariance descriptors of local features. The main contribution of this work is the use of a non-parametric statistical framework to learn the distribution of the distance between covariance descriptors and detect action changes as covariance-distance outliers.

7.1 Framework

The goal of action change detection is to partition a video into many sub-videos so that each of them contains only one single action. This may be viewed as finding temporal action boundaries inside a video, as shown in Fig. 7·1.

To reliably detect action change one must assume that the action persists for some time, i.e., actions don't keep changing at an arbitrarily fast rate. We make the assumption that actions persist for at least M frames. The main idea here is as follows. Given M frames as the dictionary of an action class, we examine the subsequent segments. If a segment is not similar to any of the segments in the dictionary, we claim that a new action class is detected (Guo et al., 2010a). The key problem here is how to measure the similarity between a segment and the dictionary. A straightforward idea is to find the nearest neighbor of this segment in the dictionary and use the NN distance as the measure of similarity. However, the issue is that an absolute threshold on the NN distance does not capture the extent of its variability

Figure 7·1: Illustration of action change boundaries.

within the same action class and would therefore not be a reliable outlier detection method. In order to solve this problem, we learn the statistical characteristics of the NN distance. If the new segment's NN distance corresponds to a low probability, it indicates that this segment comes from a new action class. The overall algorithm of action change detection is summarized as follows and illustrated in Fig. 7·2:

1. Learn the probability density function (pdf) of the NN distance between all pairs of segments in the dictionary of M training frames;

2. Compute the NN distance between a new segment and the dictionary. If it corresponds to a high probability, examine another segment by repeating this step. Otherwise, declare the detection of an action change, designate the next M frames for training and go back to the first step.

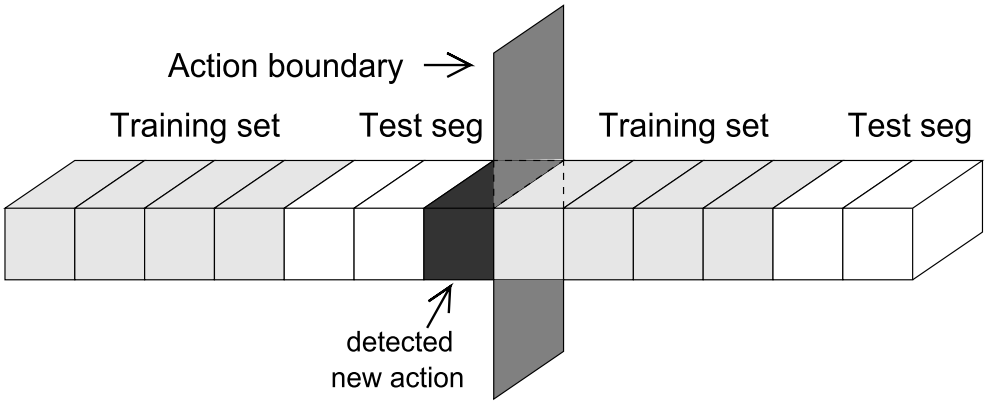

Figure 7·2: Action boundary detection approach.

7.1.1 Training phase

In the training phase, our goal is to learn the conditional pdf $p(d_{NN}|H_0)$ of the NN distance between a segment and dictionary, under the null hypothesis H_0 that this segment has the same action class as training segments in this dictionary. Let $\mathcal{F} := \{F_1, \ldots, F_k\}$ denote the set of k overlapping N-frame-long training segments forming the M-frame dictionary. For each training segment F_i we calculate d_{NN_i} the nearest distance of F_i to the remaining segments in the dictionary.

$$d_{NN_i} := \min_{j=1,\ldots,k, j\neq i} \rho(F_i, F_j). \tag{7.1}$$

Repeating this procedure k times provides k d_{NN_i} (7.1) values which can be used to estimate $p(d_{NN}|H_0)$ using a Parzen window kernel density estimation (KDE):

$$p(x|H_0) := \frac{1}{k}\sum_{i=1}^{k} \phi(x - d_{NN_i}). \tag{7.2}$$

where $\phi(x)$ is a non-negative kernel function that integrates to one.

7.1.2 Test phase

If F_t denotes a *test* action segment, we firstly compute its NN distance d_{NN}^{test} to the dictionary:

$$d_{NN}^{test} := \min_{i=1,\ldots,k} \rho(F_i, F_t).$$

Since $p(d_{NN}|H_0)$ has been estimated, we can obtain the conditional probability of d_{NN}^{test} under the null hypothesis: $p(d_{NN}^{test}|H_0)$. We can claim a detection of action change if $p(d_{NN}^{test}|H_0)$ is low. If the test segment is very similar to the dictionary, i.e., d_{NN}^{test} is very small, clearly there should be no action change. However, small values of d_{NN}^{test} are also typically associated with small $p(d_{NN}^{test}|H_0)$ values, which clearly do not correspond to the presence of an action change. This may happen if distances similar to $d_N N^{test}$ occurred infrequently during training. In order to solve this problem, we use the cumulative probability function $P(d_{NN}|H_0)$ instead of $p(d_{NN}|H_0)$:

$$P(d_{NN}|H_0) := \int_0^{d_{NN}} p(x|H_0)\mathrm{d}x. \tag{7.3}$$

We use a binary hypothesis test to decide if the test action segment is within or outside the limits of normal variability by comparing $P(d_{NN}^{test}|H_0)$ against a confidence level. If it is greater than the confidence level, then we declare an action change.

7.2 Experimental results

In order to test the performance of the proposed method, we conducted a series of experiments both on ground-truth synthetic data as well as a time-continuous camera-captured video. In each video, we applied the following pre-processing steps. First, we performed background subtraction (McHugh et al., 2009) to obtain moving object silhouettes (silhouette tunnels). We applied our action change detection algorithm with $M = 30$, i.e., we assumed a 30-frame training set. A judicious selection of this

parameter is essential for the performance of our algorithm; it must be long enough to capture salient characteristics of an action, but short enough to span a single action only. The precision of detected action boundaries depends on the length of action segments. Clearly, a large segment length increases the uncertainty of action boundary location. In our experiments, we used segments of length $N = 8$ for action comparison with a 4-frame overlap. We examined different confidence levels (0.60-0.95) in the hypothesis test.

In the ground-truth experiment we used the Weizmann dataset used earlier in 6.1 (Fig. 7·3). In order to measure the performance of our approach, we created 9 single-person multi-action test video sequences (jumping-jack, jump, sjump, run, side, skip, walk, wave1 and wave2) with exactly known action boundaries, by concatenating all the action videos in the database belonging to the same individual leaving out those action videos which have less than $M = 30$ frames. There are two types of action

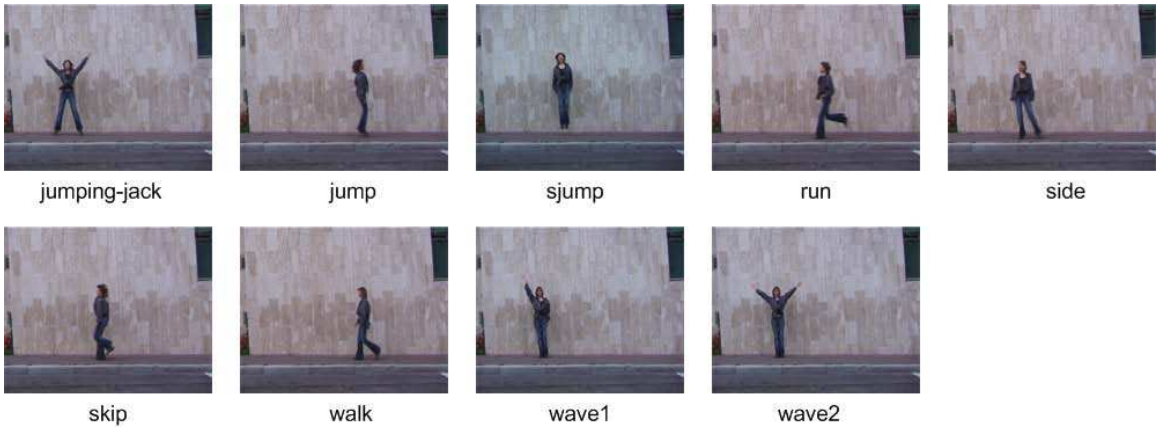

Figure 7·3: Action samples of the concatenated video sequence from Weizmann dataset.

detection errors: (i) false positive errors which occur when segments that have no action changes are classified as having action changes and (ii) false negative errors which occur when segments that have action changes are classified as having no action changes. The combined total number of action segments across all 9 test video

sequences is 597 out of which 61 segments have action changes and 536 do not. The proposed action change detection method produces 1 false negative error (Percent false negative error $P_{FN} = 100/61 = 1.64\%$) and 1 false positive error (Percent false positive error $P_{FP} = 100/536 = 0.19\%$) for a confidence level of 0.9. The values of P_{FN} and P_{FP} change by less than 1% for a range of confidence levels from 0.6–0.95. Fig. 7·4(a) shows the nearest-neighbor distances between each test segment and all segments in the immediately-preceding training set for one individual. Note that the detected change-points always correspond to large spikes in the distance measure, indicating a dissimilarity with respect to the training set. However, the nearest-neighbor distances for the detected change-points vary from 2.1 to 3.8, and this range is different for different video sequences. Finding an optimal threshold to detect the action change-points by thresholding the nearest-neighbor distance would be quite challenging. In contrast, it is clear from Fig. 7·4(b) that the cumulative probability of the nearest-neighbor distance is a more robust quantity to base the action change detection on; it adaptively adjusts the scale of nearest-neighbor distances.

In the second set of experiments, we tested our algorithm on a video sequence of single person performing 7 different actions (walk, jumping-jack, sjump, wave1, wave2, crouch and run) with natural transitions between consecutive actions, that we captured ourselves (Fig. 7·5). Although there are no ground-truth boundaries available, we subjectively identified the transitions between actions. As shown in Fig. 7·6, our algorithm has detected action change in 6 segments with index numbers: 7, 53, 92, 119, 157 and 196, exactly coinciding with the subjective transitions we had found.

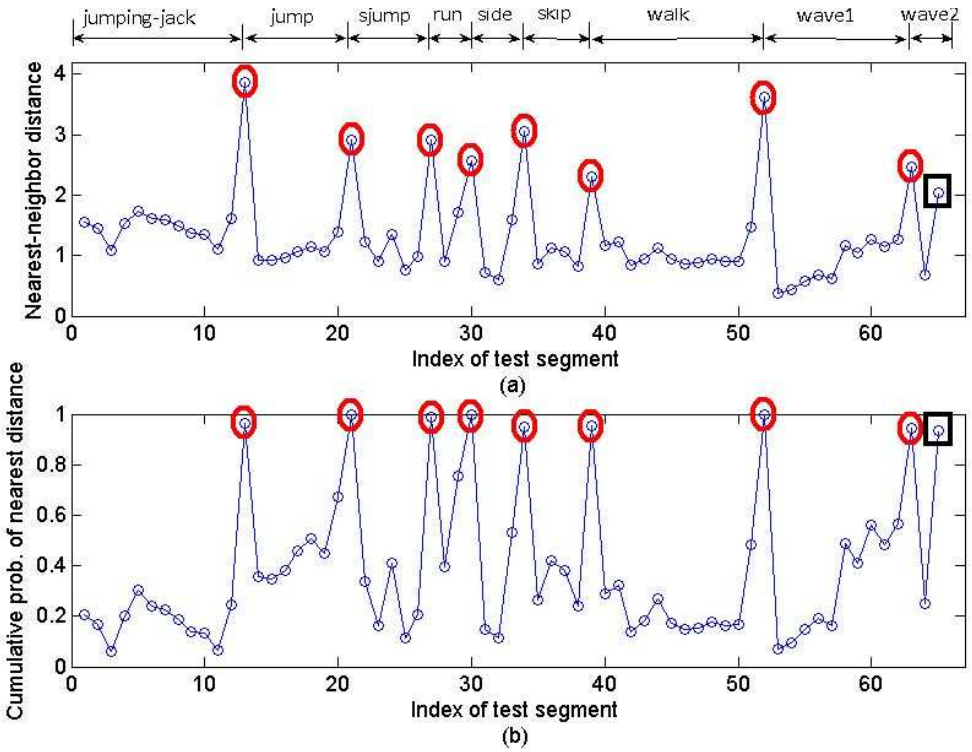

Figure 7·4: Nearest-neighbor distances (top) and corresponding cumulative probability values (bottom) for consecutive action segments from ground-truth, multi-action video sequence built from Weizmann Human Action Database for "Daria". Ground-truth actions are shown schematically above the top plot. The circles mark correctly-identified action boundaries (center of a segment in which action change takes place), while the squares mark erroneous boundaries.

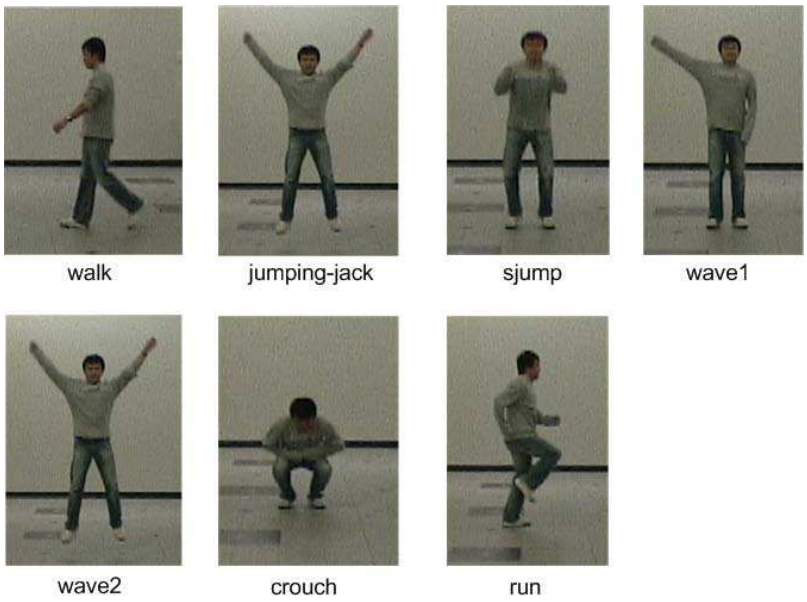

Figure 7·5: Action samples of a camera-captured video sequence.

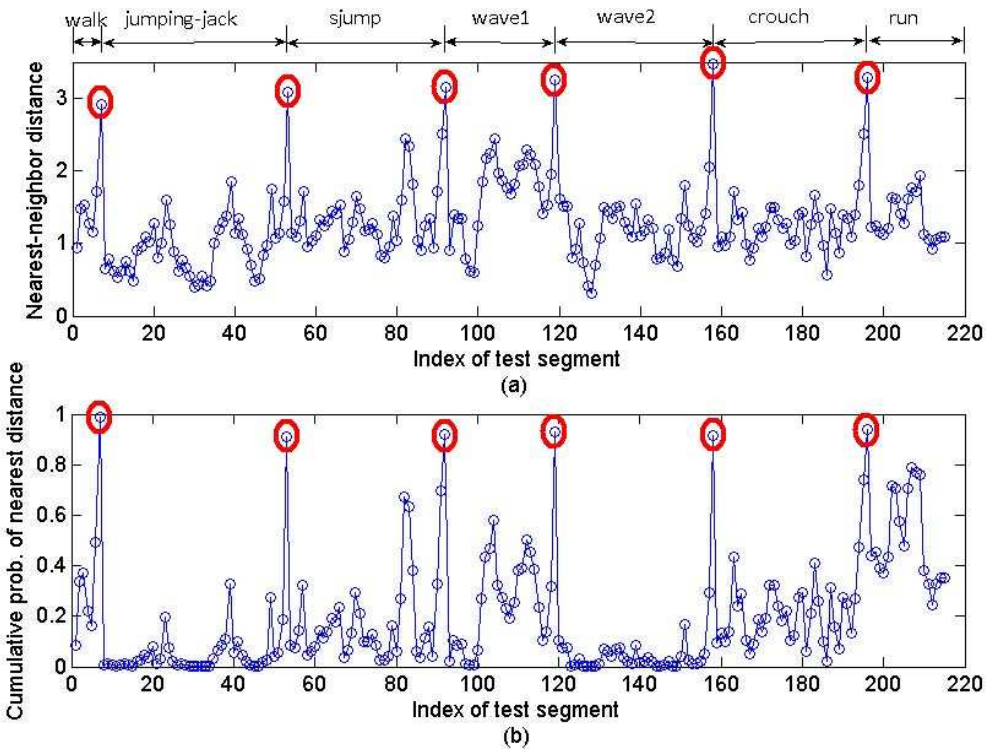

Figure 7·6: Nearest-neighbor distances (top) and corresponding cumulative probability values (bottom) for consecutive action segments from time-continuous, camera-captured video. Ground-truth actions are shown schematically above the top plot. The circles mark correctly-identified action boundaries (center of a segment in which action change takes place).

Chapter 8

Human interaction recognition

So far we have discussed how to recognize actions and how to detect action changes, based on the assumption that each action is performed by a single person. In practical scenarios, many actions actually involve multiple persons, such as kissing, hugging, handshaking, etc. We call such actions human interactions. In this chapter, we will study how to adapt our framework for recognizing human interactions, which is a challenging problem. This problem has begun to receive attention in the research community in the past decade. In Section 8.1, we review some related work in this area. In Section 8.2, we introduce the framework to solve this problem. Experimental results are shown in Section 8.3.

8.1 Overview of related work

The problem of human action interaction recognition has received attention in research community only in the last 10 years or so. (Sato and Aggarwal, 2001; Oliver et al., 2000; Aggarwal and Park, 2004; Park and Aggarwal, 2004; Park and Aggarwal, 2000; Waltisberg et al., 2010). There are two types of human interactions one is usually interested in. The first one is closely related to the relative positions, velocities and trajectories of two persons, for example, one person is following another, two people meet coming from different directions, or one person passes by another person (Sato and Aggarwal, 2001; Oliver et al., 2000). All these interactions are composed of walking, running and standing. The only differences are due to different motion

trajectories, relative velocities and positions. The second type of human interactions involves more detailed information about an individual's action, such as handshaking, hugging, pushing, etc (Aggarwal and Park, 2004; Park and Aggarwal, 2004; Park and Aggarwal, 2000; Waltisberg et al., 2010). Trajectories, velocities and positions are only secondary factors here, since each interaction is composed of different individual actions. In this thesis, we only focus on the second type of human interaction.

Sato *et al.* (Sato and Aggarwal, 2001) proposed a method to recognize the first type of human interactions. It consists of two major components: trajectory extraction (human segmentation and tracking) and trajectory classification (interaction classification). The trajectory extraction process outputs human trajectories as the physical positions of humans in the scene over time by recognizing and tracking human images. The interaction classification process extracts some features, such as relative distance and velocity of each person, from the trajectory shape. Then, it selects the most likely behavior type in the training dataset based on feature similarity. Oliver *et al.* (Oliver et al., 2000) proposed a Bayesian system for modeling the first type of human interactions. They use a Kalman filter to track an individual's position, velocity and coarse shape, and then model the relationship between individuals by using the coupled Hidden Markov Model (CHMM).

In order to recognize the second type of interactions, Park *et al.* (Park and Aggarwal, 2004) developed a method using a hierarchical Bayesian network (BN). First, body parts are tracked and their poses are estimated at a low level of BN. The overall body pose is estimated at a high level of BN. The pose estimation results are then concatenated to form a sequence, and sequence classification is performed by a dynamic Bayesian network. The recognition of two-person interactions is expressed in terms of semantic verbal descriptions at multiple levels: individual body-part motions at low level, single-person actions at middle level, and two-person interactions at

the highest level. Park *et al.* (Park and Aggarwal, 2000) also proposed a method based on the nearest neighbor classifier applied to a parametric human-interaction model, which describes the interpersonal relationship. Waltisberg *et al.* (Waltisberg et al., 2010) proposed a hough-voting action recognition system that recognizes each individual's action separately and then combines the classification results to obtain an action label of the interaction.

8.2 Proposed framework

In previous section, we reviewed state-of-art methods for human interaction recognition. In this section, we propose a new framework, which involves individual action recognition followed by a combination of individual action labels. There are basically two approaches for human interaction recognition: the universal approach and divide-and-conquer approach.

The universal approach treat interacting persons as a single object with multiple disconnected parts. The advantage of this approach is that it is simple and can directly fit into our previous action recognition framework in Chapter 3 without concerning how many moving persons are involved in the human interaction. However, as we will see, this approach would require more training samples to well represent human interactions.

The divide-and-conquer approach, in contrast, divides video spatially to isolate individuals, analyze them separately, and fuse the results. This approach is more complicated since we need object tracking to isolate moving individuals and analyze each action separately. The advantage is that it needs less training samples to well represent human interactions compared with the universal approach. Consider the following example: a human interaction e.g. punching involves two individuals - an attacker who is hitting and a defender who is dodging. If we assume that there are

10 action variations for each individual, then the total number of possible realizations of punching is $10 \times 10 = 100$. In other words, the universal approach needs 100 training samples to well represent punching in this example. In contrast, the divide-and-conquer approach only needs $10 + 10 = 100$ samples to well represent the interaction. In general, if there are multiple individuals involved in a human interaction and M_k denotes the number of training samples needed for reliably recognizing the k-th action in the interaction, then the number of examples of human interaction needed to reliable recognize the interaction would be $N_1 = \prod_k M_k$. Since N_1 is at least an order of magnitude larger than M_k, it is much more difficult to obtain sufficient training samples for interaction, compared to an individual's action. This motivates us to recognize human interaction by dividing it into individuals and classifying each of them separately. Then, we only need M_k training samples to recognize each individual's action, and thus the sufficient number of training samples for recognizing human interaction is vien by $N_2 = \sum_k M_k$, which is at least an order of magnitude less than N_1.

Assume that we have divided a human interaction sequence into individual sequences. Each individual's action label can be obtained based on our previous approach described in Chapter 3. Then, a human interaction's label can be obtained based on individuals' actions. Ideally, if each individual is correctly recognized, the interaction will be correctly recognized as well. However, in practice, an individual may be misclassified, which may make the decision fusion ambiguous. Consider the previous punching example. If the defender's action is misclassified as handshaking, then the label combination of hitting and handshaking will make no sense. We can, of course, treat such confused decisions as misclassifications. The outcome is that our method will be sensitive to individual errors. In other words, any individual's misclassification will lead to the misclassification of the interaction. A better alternative

is to estimate confidence measures for each individual, that reflects the confidence of each action class to be the best class. Then, in the decision fusion step, we can combine the confidence measures to obtain an interaction label. A hard fusion , i.e., combining labels, only make use of the best action class of each individual. On the contrary, a soft fusion, i.e., combining confidence measures, merges more information (not only the individual's best-matched class but also the confidence of each action class), and can effectively resolve ambiguities occurring in hard fusion.

The overall framework of our approach is as follows, as also shown in Fig. 8·1.

1. Divide a human interaction video into separate individuals by using an object tracking algorithm;

2. Estimate confidence measures for each individual that reflects the confidence of each action class to be the best class.

3. Combine the confidence measures to obtain the interaction label.

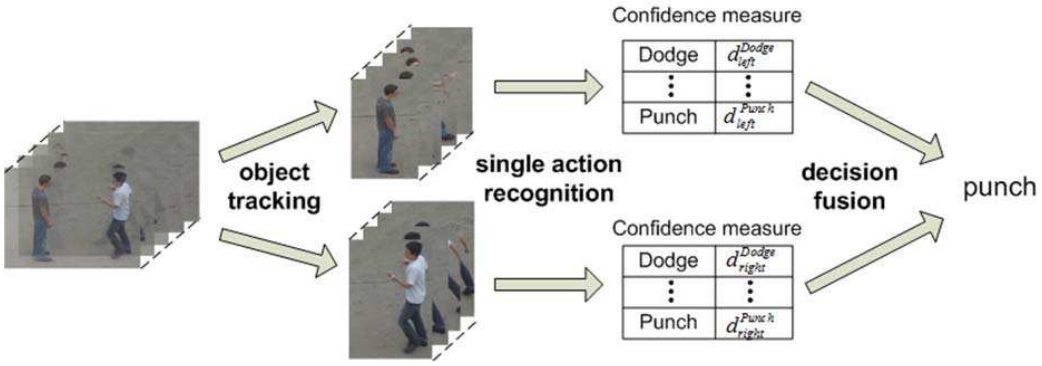

Figure 8·1: Proposed framework for human interaction recognition.

8.2.1 Individual action recognition

We now discuss how to compute the confidence measures given a query individual action. The distance between query video and each action class is apppropriate

to indicate the confidence of each class. The smaller the distance, the higher the confidence, and vice versa. Now, the question is how to define the distance between query video and each action class. Given a query video, we can partition it into overlapping segments. Then, for each query segment, we find its nearest neighbor segment among all training segments in action class k and record the NN distance. Finally, after obtaining NN distances of all query segments, we take the average and use this mean distance as the distance between query video and action class k, as shown in Fig. 8·2. More formally, let C_i^q be the covariance matrix of ith query segment (there are N query segments in the query video) and let $C_{k,j}$ be the covariance matrices of training segments in action class k. The mean distance d_k between query video and each action class can be expressed as:

$$d_k = \frac{1}{N} \sum_i \min_j \rho(C_i^q, C_{k,j}), \tag{8.1}$$

where ρ is the affine-invariant Riemannian metric (3.2) or the log-Euclidean metric (3.4) between covariance matrices. So far we have assumed that all the segments have the same length (number of frames). Actually, segment length may include useful information to discriminate different actions. The fixed segment length can be treated as a special case of variable segment length. Suppose we have obtained a series of appropriate segment lengths for all training classes: $L_1, L_2, \cdots, L_K$ (K is the number of action classes). Then, segment length of each class is L_k. In order to measure the distance between query video and each training class, we need to partition the query video into K segment sets according to K segment lengths. For the kth set of query segments, we can measure the distance between query video and the kth training class using (8.1), shown in Fig. 8·3. C_i^{q,L_k} is the covariance matrix of ith query segment based on segment length L_k, d_i^{NN,L_k} is the NN distance between the ith query segment and its nearest neighbor in training class k, and d^{L_k} is the

distance between query video and kth training class. The use of variable segment length increases the flexibility of training video partitioning. The success of this method relies on the appropriate selection of segment lengths for each action class. However, it seems difficult to develop a systematic approach to find the optimal variable segment lengths for different classes, and we use a heuristic to manually adjust the segment lengths. The basic rule of our manual adjustment is that if class i and class j are confused in classification, the difference between their corresponding segment lengths should be increased.

8.2.2 Decision fusion

Once we obtain the confidence measures of each individual in a human interaction video, the next step is to determine the action label of this interaction by fusing the confidence measures. We need to pay attention to two observations:

- Individual action classes and human interaction action classes may not be exactly the same. Take the punching as an example again, the action *dodging* is not an interaction action.

- Some combinations of individual actions are inconsistent with each other. For example, the combination of handshaking and punching is not a valid interaction.

The first observation above suggests that we may need to introduce extra action classes for recognizing an individual action, e.g., dodging. The second observation suggests that we should only focus on those combinations of individual actions that make sense when performing the decision fusion. If there are K individual action classes, the possible number of combinations is K^2, but the actual number of interaction that are consistent is usually much lower than K^2.

The process of decision fusion is shown in Fig. 8·4. For each real human interaction, find its individual actions' distance measures in the ranking lists. Then, the mean of these distances indicates the distance between the query interaction video and this interaction class. We select the interaction class that is closest to the query video.

Our approach to human interaction recognition has unique properties, compared with our single-person recognition method. At the segment level, it measures the distance between a query segment and each training class, rather than determining the label of this segment. At the sequence level, it uses "soft fusion" rather than "hard fusion"(majority rule) to label the query video. The experimental results are presented in the next section.

8.3 Experimental results

In order to test the performance of our method, we conducted some experiments on the so-called UT-interaction dataset. This dataset was used at the "High-level Human Interaction Recognition Challenge" in the ICPR 2010 Contest on Semantic Description of Human Activities (SDHA). The UT-Interaction dataset contains videos of continuous executions of 6 classes of human interactions: shaking hands, pointing, hugging, pushing, kicking and punching. Several participants in more than 15 different types of clothing appear in the videos. The videos are taken at the resolution of 720×480, 30fps, and the height of a person in the video is about 200 pixels. This dataset is challenging, because pair of individuals performs each action once. In other words, all actions in this dataset are non-repetitive. Therefore, we do not have a sufficient number of samples to represent each action. Additionally, we introduce one more individual action of *dodging*, which appears in kicking, punching and pushing. Thus, there are 7 individual actions in total. We tested the interaction recognition

performance based on both fixed segment length and variable segment length. The fixed segment length we chose was 20. We manually selected the variable segment lengths follows: 10,14,10,16,8,20,10. In our tests, we used feature covariance matrices based on optical flow, and LOOCV. Table 8.1 shows the confusion matrix based on fixed-length segments. We see that punching and pushing, kicking and pushing, hugging and handshaking are difficult to discriminate. Thus, we adjusted the segment lengths so that these actions' segment lengths are quite different. Table 8.2 shows the confusion matrix based on variable-length segments, which improves the performance by more than 20%. Table 8.3 shows results from Waltisberg *et al.* (Waltisberg et al., 2010). The performance of our approach is in line with state-of-the-art methods today.

Table 8.1: Interaction confusion matrix obtained by the proposed approach based on fixed length segments (SEQ-CCR = 61.67%).

	handshake	hug	kick	point	punch	push
handshake	9	1	0	0	0	0
hug	4	6	0	0	0	0
kick	0	0	5	1	1	3
point	3	0	1	5	1	0
punch	0	0	2	0	5	3
push	0	0	1	0	2	7

Table 8.2: Interaction confusion matrix obtained by the proposed approach based on variable length segments (SEQ-CCR = 85.0%).

	handshake	hug	kick	point	punch	push
handshake	10	0	0	0	0	0
hug	0	10	0	0	0	0
kick	0	1	7	0	2	0
point	2	0	2	6	0	0
punch	0	0	0	0	10	0
push	0	2	0	0	0	8

Table 8.3: Interaction action confusion matrix obtained by the method of Waltisberg *et al.* (Waltisberg et al., 2010) (SEQ-CCR = 88.33%).

	handshake	hug	kick	point	punch	push
handshake	7	2	1	0	0	0
hug	0	10	0	0	0	0
kick	0	0	10	0	0	0
point	0	0	0	10	0	0
punch	0	0	2	0	7	1
push	0	0	1	0	0	9

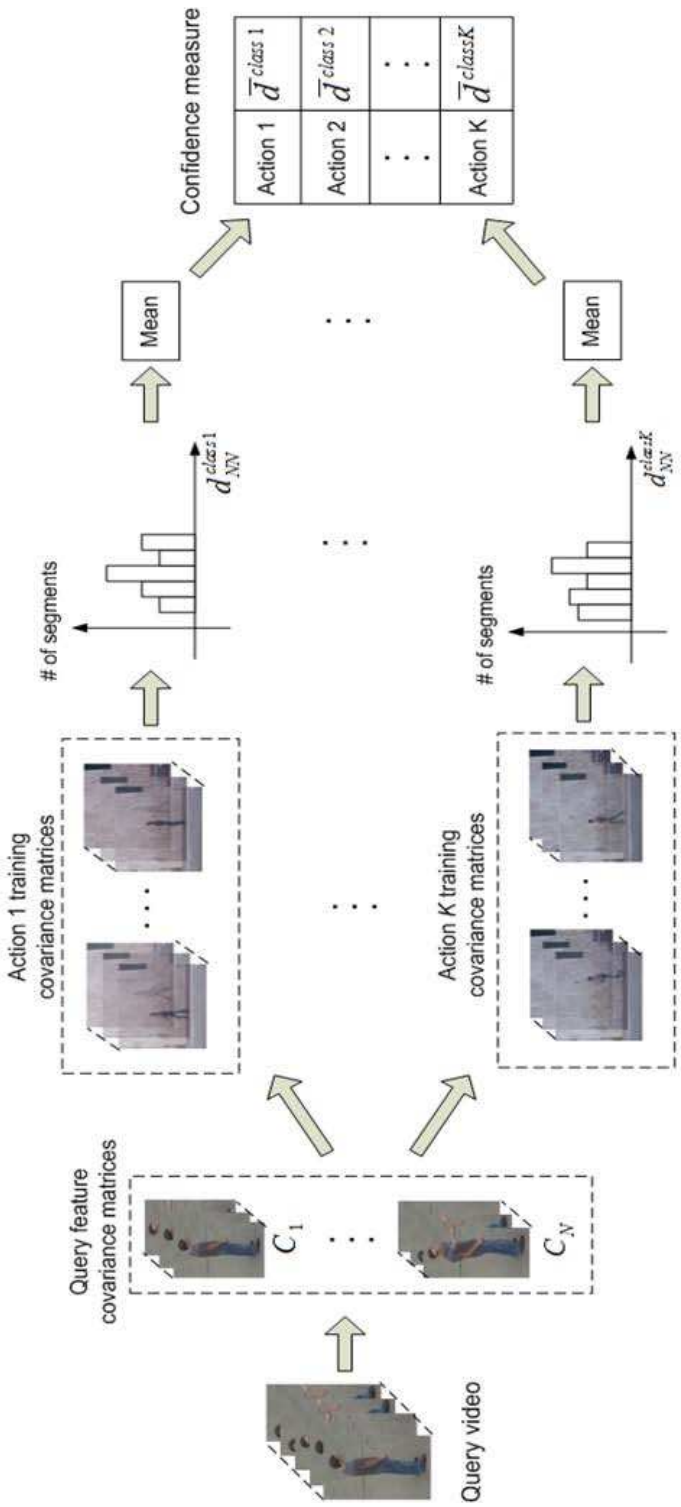

Figure 8·2: Computation of confidence measures based on fixed segment length.

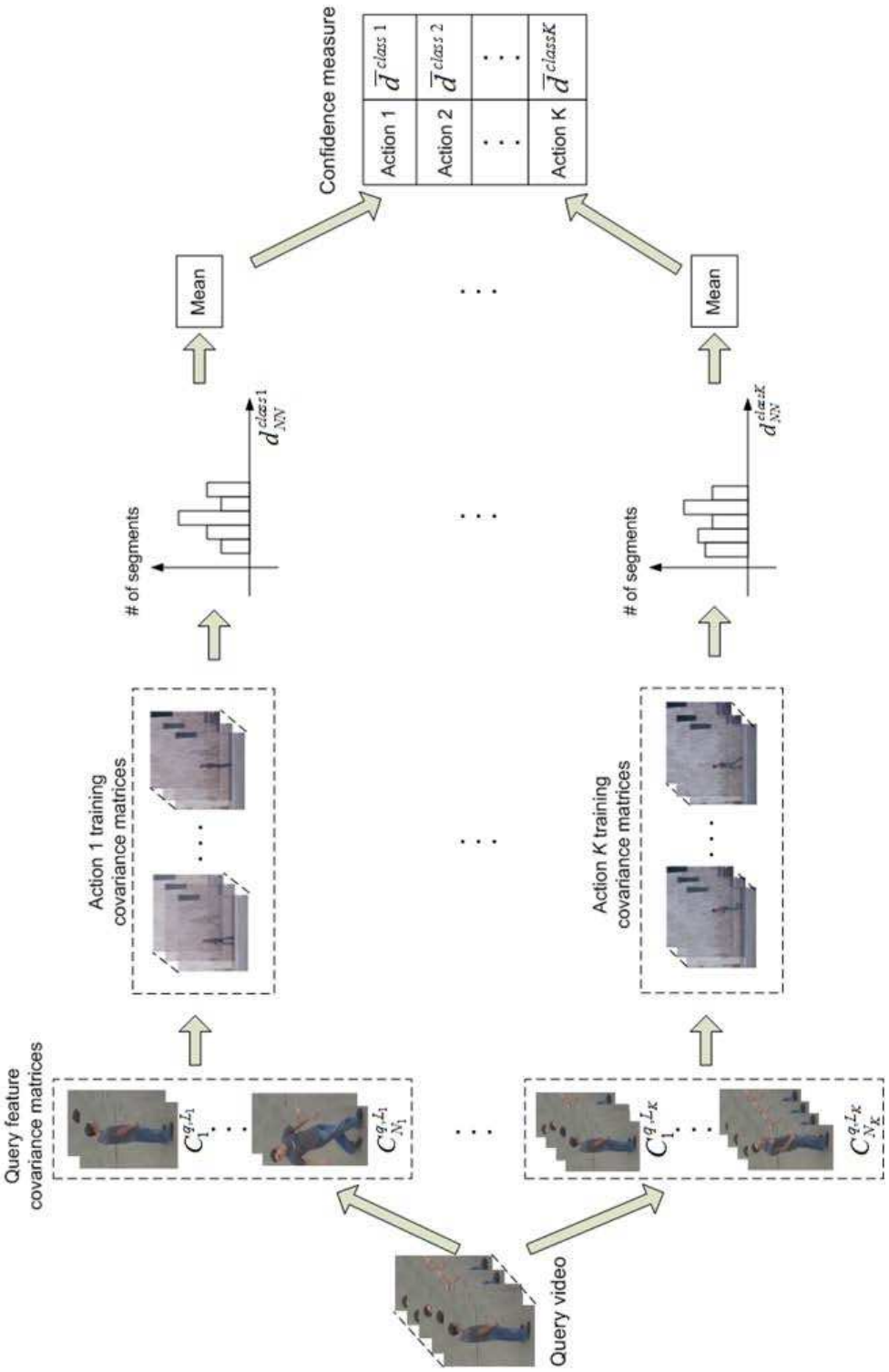

Figure 8·3: Computation of confidence measures based on variable segment length.

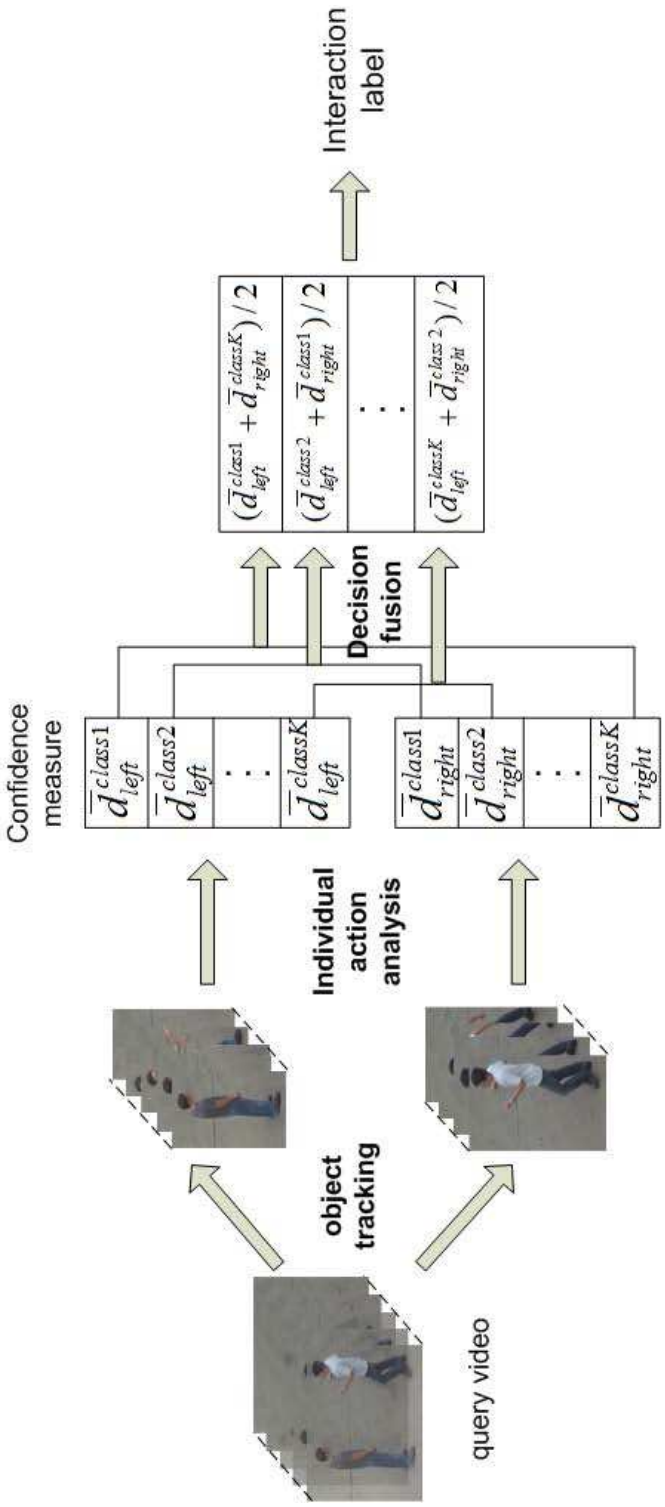

Figure 8·4: Flow chart illustrating the process of decision fusion in human interaction recognition.

Chapter 9

Conclusions and future work

This chapter summarizes this thesis, and proposes ideas for future work. The main inspiration for this work was the rapid development of content-based video analysis. In the past decades, many researchers have been working in video processing. Most of the work has focused on low-level processing, such as motion detection, video segmentation, object tracking, etc. Although low-level processing can detect motion and isolate moving objects, it cannot classify motion. Thus, it is desirable to develop automatic methods to analyze video data with the goal of understanding the visual environment. Action recognition is a content-based technique that can provide semantic information about a video. It can lead to many applications, e.g., video surveillance, video search and human-computer interaction. The action recognition task is made complex by the complexity of the scene (multiple interacting moving objects, clutter, occlusions, illumination variability, etc.), the camera (imperfections, motion and shake, and viewpoint), and the complexity of actions (non-rigid objects and intra- and inter-class action variability).

9.1 Summary of contributions

In this work, we studied a sub-problem of action recognition in which there is only one action in a video performed by an individual. We proposed a systematic action recognition framework, including action representation and action classification. An action is represented by log-covariance matrix of a bag of local features. These features

are associated with moving pixels, providing a coarse localized description of the action. Feature covariance matrices provide a compact representation since they extract second order statistics of local features and lie in a much lower dimensional space than the local features. In order to be effective, the selected local features should involve discriminative properties of motion dynamics. There are two examples of local features in this thesis, those based on silhouette tunnel and those based on optical flow. Silhouette tunnel describes the shape of an action and optical flow describes the motion dynamics of an action. Both of them include important information for classifying human actions.

In action classification, we used two supervised learning classifiers: the nearest neighbor (NN) and sparse linear approximation (SLA) classifiers. In the SLA classifier, we approximate the logarithm of a query action covariance matrix by a sparse linear combination of the logarithm of training action covariance matrices using a fast linear program and determine the action label from sparse coefficients. Common to both classifiers is the novel idea that classification algorithms that have been developed for vectors can be re-purposed for covariance tensors by using a log-nonlinearity to map the convex cone of covariance matrices to the vector space of symmetric matrices.

We also demonstrated how our approach can be used for sequentially detecting changes in actions in an adaptive unsupervised manner so as to parse a long video sequence into sub-sequences, each of which only includes a single action class. Then, we can employ our proposed approach to get action label of each sub-sequence. We used a non-parametric statistical framework to learn the distribution of the nearest-neighbor Riemannian distances between feature covariance matrices of video segments. Then, we used binary hypothesis testing to determine if new video segments include action changes.

We also proposed a method how to recognize human interactions, which is usually a more challenging problem than single action recognition. Given an interaction video, first we spatially divide it into individual actions using object tracking. Then, we estimated a ranked list for each individual. Each entry in this list reflects the chance of each action class to be the best class. Then, in the decision fusion step, we combined the ranked lists to obtain the interaction label. We tested the performance on UT-Tower interaction dataset, and our method turned out to perform very closely to state-of-the-art algorithms.

Overall the contributions of this thesis can be summarized as follows:

- We developed a new action recognition framework based on log-covariance matrices of bags of local action features;
- We generalized the vector-based NN and SLA Classification algorithms to covariance matrices;
- We discovered discriminative features for action recognition from object silhouette tunnels and optical flow;
- We extended the feature covariance matrix framework to the unsupervised action change detection problem;
- We developed a feature covariance based method for human interaction recognition.

9.2 Future work

This section briefly describes a few directions that could extend our work.

9.2.1 Alternative features

In this thesis, we introduced two examples of local features based on silhouette tunnel and optical flow. However, there may exist more discriminative features that can better represent motion dynamics. Recently, we have adapted our action recognition framework to perform real-time gesture recognition using the Kinect camera from Microsoft. Kinect provides robust and reliable skeleton model of a person in front of it, and we used parameters of this model as features for gesture recognition. Our action recognition framework proved very useful even in this very simple scenario of a skeleton model. Clearly, our action recognition framework is general, and we believe that the performance of our approach can be further improved if better features are judiciously selected.

9.2.2 Group action recognition

Here, we primarily focused on the problem in which there is one or two moving persons in a video. In practice, a camera usually captures several people simultaneously. Due to occlusions and low resolution of individuals, it is difficult to recognize a group action. Additionally, it is also challenging to recognize if there is an unusual action in a crowded scene. These are practical issues that arise from video surveillance applications, which are worthwhile to study and resolve.

9.2.3 Extension of current assumptions

In Chapter 1 we listed challenges associated with action recognition and made some assumptions in order to make our problem tractable. However, some assumptions are not practical and make our approach difficult to adapt to real-life scenarios. Thus, in the future, action recognition should be studied without constraining assumptions, such as clutter, occlusions, camera motion, zoom-in/zoom-out and camera viewpoint change. All these are practical issues that prevent us from accurately recognizing

actions.

The datasets we have focused on are limited. In the future, it would be desirable to work on more challenging datasets, such as the VIRAT dataset (Oh et al., 2011). This dataset is designed to be realistic, natural and challenging for action recognition in terms of camera motion, background clutter and diversity in the recorded scenes. There are frequent incidental movers and background activities in videos. This is a very challenging dataset that relaxes most of our assumptions and thus requires more effort to achieve satisfactory action recognition performance.

9.2.4 Action recognition via static scenes

Our approach has only studied action recognition by focusing on the moving person. Sometimes, the scene and individual objects in videos can also help understand actions. For example, if we can recognize a swimming pool in a video, it is likely that persons in this video are swimming. Action characteristics are not the only factor that can help us recognize it. However, recognizing scene or objects in a video is still very challenging, perhaps even more challenging than recognizing actions. This approach may be useful in scenarios where we can easily recognize scenes and objects.

References

Aggarwal, J. and Park, S. (2004). Human motion: Modeling and recognition of actions and interactions. In *Proceedings of 2nd International Symposium on 3D Data Processing, Visualization and Transmission*, pages 640–647.

Ahmad, M., Parvin, I., and Lee, S. W. (2010). Silhouette history and energy image information for human movement recognition. *Journal of Multimedia*, 5(1):12.

Ali, S. and Shah, M. (2010). Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):288–303.

Arsigny, V., Pennec, P., and Ayache, X. (2006). Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine*, 56(2):411–421.

Basseville, M. and Nikiforov, I. (1993). *Detection of abrupt changes: theory and application*, volume 15. Citeseer.

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

Brand, M. and Kettnaker, V. (2000). Discovery and segmentation of activities in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):844–851.

Chen, C. C., Ryoo, M. S., and Aggarwal, J. K. (2010). UT-Tower dataset: Aerial view activity classification challenge. http://cvrc.ece.utexas.edu/sdha2010/aerial_view_activity.html.

Chen, Y., Wu, Q., and He, X. (2008). Human action recognition by radon transform. In *Proceedings of the IEEE International Conference on Data Mining Workshops*, pages 862–868.

Collins, R. T., Gross, R., and Shi, J. (2002). Silhouette-based human identification from body shape and gait. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 366–371.

Concha, O., Da, X., and Piccardi, M. (2010). Compressive sensing of time series for human action recognition. In *Proceedings of the IEEE International Conference on Digital Image Computing: Techniques and Applications*, pages 454–461.

Cunado, D., Nixon, M. S., and Carter, J. N. (2003). Automatic extraction and description of human gait models for recognition purposes. *Computer Vision and Image Understanding*, 90(1):1–41.

Danafar, S. and Gheissari, N. (2007). Action recognition for surveillance applications using optic flow and svm. *Asian Conference on Computer Vision*, pages 457–466.

Dollar, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *Proceedings of the IEEE International Conference on Visual Surveillance and Performance Evaluation of Tracking and Surveillance Workshop*, pages 65–72.

Duda, R., Hart, P., Stork, D., et al. (2001). *Pattern classification*, volume 2. Wiley New York.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745.

Elgammal, A., Duraiswami, R., Harwood, D., and Davis, L. (2002). Background and foreground modeling using nonparametric kernel density for visual surveillance. *Proceedings of the IEEE*, 90:1151–1163.

Fathi, A. and Mori, G. (2008). Action recognition by learning mid-level motion features. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Forstner, W. and Moonen, B. (1999). A metric for covariance matrices. *Techical Report*. Department of Geodesy and Geoinformation, Stuttgart University.

Goncalves, L., Bernardo, E. D., Ursella, E., and Perona, P. (1995). Monocular tracking of the human arm in 3d. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 764–770.

Gorelick, L., Blank, M., Shechtman, E., Irani, M., and Basri, R. (2007). Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253.

Guo, K., Ishwar, P., and Konrad, J. (2009). Action recognition from video by covariance matching of silhouette tunnels. In *Proceedings of Brazilian Symposium on Computer Graphics and Image Processing*.

Guo, K., Ishwar, P., and Konrad, J. (2010a). Action change detection in video by covariance matching of silhouette tunnels. In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, pages 1110–1113.

Guo, K., Ishwar, P., and Konrad, J. (2010b). Action recognition in video by sparse representation on covariance manifolds of silhouette tunnels. *Recognizing Patterns in Signals, Speech, Images and Videos*, pages 294–305.

Guo, K., Ishwar, P., and Konrad, J. (2010c). Action recognition using sparse representation on covariance manifolds of optical flow. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 188–195.

Hoai, M., Lan, Z., and Torre, F. (2011). Joint segmentation and classification of human actions in video. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3265–3272.

Ikizler, N. and Duygulu, P. (2007). Human action recognition using distribution of oriented rectangular patches. *Human Motion–Understanding, Modeling, Capture and Animation*, pages 271–284.

Kale, A., Sundaresan, A., Rajagopalan, A. N., Cuntoor, N. P., Roy-Chowdhury, A. K., Kruger, V., and Chellappa, R. (2004). Identification of humans using gait. *IEEE Transactions on Image Processing*, 13(9):1163–1173.

Ke, Y., Sukthankar, R., and Hebert, M. (2005). Efficient visual event detection using volumetric features. In *Proceedings of the IEEE International Conference on Computer Vision*.

Kovashka, A. and Grauman, K. (2010). Learning a hierarchy of discriminative space-time neighborhood features. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learing realistic human actions from movies. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Le, Q., Zou, W., Yeung, S., and Ng, A. (2011). Learning hierarchical invariant spatio-temporal features for action recogition with independent subspace analysis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Liu, J., Ali, S., and Shah, M. (2008). Recognizing human actions using multiple features. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Liu, J., Luo, J., and Shah, M. (2009). Recognizing realistic actions from videos in the wild. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1996–2003.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision*, page 1150.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Mahoor, M., Zhou, M., Veon, K., Mavadati, S., and Cohn, J. (2011). Facial action unit recognition with sparse representation. In *Proceedings of the IEEE International Conference on Automatic Face Gesture Recognition and Workshops*, pages 336–342.

Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. (2008). Discriminative learned dictionaries for local image analysis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

McHugh, J., Konrad, J., Saligrama, V., and Jodoin, P.-M. (2009). Foreground-adaptive background subtraction. *IEEE Signal Processing Letter*, 16(5):390–393.

Mei, X. and Ling, H. (2009). Robust visual tracking using l1 minimization. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Niebles, J., Wang, H., and Fei-Fei, L. (2008). Unsupervised learning of human action categories using spatial-temporal words. In *International Journal of Computer Vision*.

Oh, S., Hoogs, A., Perera, A., Cuntoor, N., Chen, C., Lee, J., Mukherjee, S., Aggarwal, J., Lee, H., Davis, L., et al. (2011). A large-scale benchmark dataset for event recognition in surveillance video. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, page 6.

Oliver, N. M., Rosario, B., and Pentland, A. P. (2000). A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843.

Park, S. and Aggarwal, J. (2000). Recognition of human interaction using multiple features in gray scale images. In *Proceedings of 15th International Conference on Pattern Recognition*, volume 1, pages 51–54.

Park, S. and Aggarwal, J. (2004). A hierarchical bayesian network for event recognition of human actions and interactions. *Multimedia Systems*, 10(2):164–179.

Raptis, M., Wnuk, K., and Soatto, S. (2010). Spike train driven dynamical models for human actions. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Ristivojević, M. and Konrad, J. (2006). Space-time image sequence analysis: object tunnels and occlusion volumes. *IEEE Transactions on Image Processing*, 15(2):364–376.

Rohr, K. (1994). Towards model-based recognition of human movements in image sequences. *Computer Vision Graphical Models and Image Processing -Image Understanding*, 59(1):94–115.

Sato, K. and Aggarwal, J. K. (2001). Tracking and recognizing two-person interactions in outdoor image sequences. In *Proceedings of the IEEE International Workshop on Multi-Object Tracking*, page 87.

Schuldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: A local svm approach. In *International Conference on Pattern Recognition*.

Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th International Conference on Multimedia*, page 360. ACM.

Seo, H. J. and Milanfar, P. (2011). Action recognition from one example. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):867–882.

Shi, Q., Wang, L., Cheng, L., and Smola, A. (2008). Discriminative human action segmentation and recognition using semi-markov model. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Smith, P., da Vitoria Lobo, and Shah, M. (2005). Temporal boost for event recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1.

Starner, T. and Pentland, A. (1995). Visual recognition of american sign language using hidden markov model. In *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*.

Tuzel, O., Porikli, F., and Meer, P. (2006). Region covariance: A fast descriptor for detection and classification. In *Proceedings of European Conference on Computer Vision*.

Tuzel, O., Porikli, F., and Meer, P. (2008). Pedestrian detection via classification on Riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1713–1727.

Waltisberg, D., Yao, A., Gall, J., and Gool, L. V. (2010). Variations of a hough-voting action recognition system. *Recognizing Patterns in Signals, Speech, Images and Videos*, pages 306–312.

Wang, H., Klaser, A., Schmid, C., and Liu, C. (2011). Action recognition by dense trajectories. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Wang, L., Ning, H., Tan, T., and Hu, W. (2004). Fusion of static and dynamic body biometrics for gait recognition. *IEEE Transactions on Circuits System of Video Technology*, 14(2):149–158.

Wang, X., Ma, X., and Grimson, W. (2009). Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):539–555.

Wang, Y., Huang, K., and Tan, T. (2007). Human activity recognition based on r transform. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Wang, Y. and Mori, G. (2009). Human action recognition by semi-latent topic models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1762–1774.

Wong, S. F. and Cipolla, R. (2007). Extracting spatio-temporal interest points using global information. In *Proceedings of the IEEE International Conference on Computer Vision*.

Wright, J., Yang, A., Ganesh, A., Sastry, S., and Ma, Y. (2009). Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227.

Wu, X., Xu, D., Duan, L., and Luo, J. (2011). action recognition using context and appearance distribution features. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Xiang, T. and Gong, S. (2006). Beyond tracking: modeling activity and unstangding behaviour. *International Journal of Computer Vision*, 67(1):21–51.

Xie, Y., Chang, H., Li, Z., Liang, L., Chen, X., and Zhao, D. (2011). A unified framework for locating and recognizing human actions. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time sequential image using hidden markov model. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Yang, J., Wright, J., Huang, T., and Ma, Y. (2008). Image super-resolution as sparse representation of raw image patches. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Yilmaz, A. and Shah, M. (2005). Action sketch: A novel action representation. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Ying, Z., Wang, Z., and Huang, M. (2010). Facial expression recognition based on fusion of sparse representation. *Advanced Intelligent Computing Theories and Applications with Aspects of Artificial Intelligence*, pages 457–464.

Zach, C., Pock, T., and Bischof, H. (2007). A duality based approach for realtime tv-l1 optical flow. In *Pattern Recognition*, pages 214–223.

Zhang, T., Liu, J., Ouyang, Y., and Lu, H. (2009). Boosted Exemplar Learning for Human Action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*.

CURRICULUM VITAE

Kai Guo

Contact Information

Address: ECE Department 8 St. Mary Street Boston MA 02215

Email: kaiguo@bu.edu

Web: blogs.bu.edu/kaiguo

Education

Ph.D., Boston University, 2011

Department of Electrical and Computer Engineering

M.Phil., City University of Hong Kong, 2007

Department of Electronic Engineering

B.S., Xidian University, 2005

Department of Electronic Engineering

Honors

- Best Paper Award in IEEE Int. Conf. Advanced Video and Signal-based Surveillance, Boston, 2010.
- Winner of the "Aerial-view Activity Recognition Challenge" in the *International Conference on Pattern Recognition*, Turkey, 2010
- Dean's Fellowship, College of Engineering, Boston University, 2007-2008
- Cheung Family Fellowship, Boston University, 2007-2008
- Graduate Student Scholarship of City University of Hong Kong 2007
- Excellent Student Award, Xidian University, 2002-2005
- The First Grade Scholarship Award, Xidian University, 2002-2005

Publications

- K. Guo, P. Ishwar and J. Konrad, "Action recognition on covariance manifold", to be submitted.

- K. Guo, P. Ishwar and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow", in *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 188-195, Boston, 2010.

- K. Guo, P. Ishwar and J. Konrad, "Action recognition in video by sparse representation on covariance manifolds of silhouette tunnels", in *Recognizing Patterns in Signals, Speech, Images and Videos*, pages 294-305, Turkey, 2010.

- K. Guo, P. Ishwar and J. Konrad, "Action change detection in video by covariance matching of silhouette tunnels.", in *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, pages 1110-1113, Dallas, 2010.

- K. Guo, P. Ishwar and J. Konrad, "Action recognition from video by covariance matching of silhouette tunnels.", in *Proceedings of Brazilian Symosium on Computer Graphics and Image Processing*, pages 299-306, Brazil, 2009.

- L.M.Po and K. Guo, "Transform-domain fast sum of the squared difference computation for H.264 rate-distortion optimization", *IEEE Transactions on Circuits System of Video Technology*, vol. 17, No. 6, page 765-773, 2007.

- K. Guo and L.M.Po, "A new partial codeword updating scheme based on rate-distortion optimization for adaptive vector quantization", *in Proceedings of 2007 International Conference on Multimedia and Expo*, page 751-754, China, 2007.