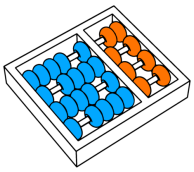
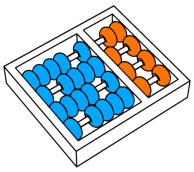


Thierry Pinheiro Moreira

"Real-Time Human Action Recognition Based on Motion Shapes"

"*Reconhecimento de Ações Humanas em Tempo Real Baseado em Figuras de Movimento*"

CAMPINAS
2014



University of Campinas
Institute of Computing

Universidade Estadual de Campinas
Instituto de Computação

Thierry Pinheiro Moreira

"Real-Time Human Action Recognition Based on Motion Shapes"

Supervisor:
Orientador: **Prof. Dr. Hélio Pedrini**

"Reconhecimento de Ações Humanas em Tempo Real Baseado em Figuras de Movimento"

MSc Dissertation presented to the Post Graduate Program of the Institute of Computing of the University of Campinas to obtain a Master degree in Computer Science.

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Ciência da Computação.

This volume corresponds to the final version of the Dissertation defended by Thierry Pinheiro Moreira, under the supervision of Prof. Dr. Hélio Pedrini.

Este exemplar corresponde à versão final da Dissertação defendida por Thierry Pinheiro Moreira, sob orientação de Prof. Dr. Hélio Pedrini.

Hélio Pedrini

Supervisor's signature / *Assinatura do Orientador*

CAMPINAS
2014

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Maria Fabiana Bezerra Muller - CRB 8/6162

M813r

Moreira, Thierry Pinheiro, 1990-
Real-time human action recognition based on motion shapes / Thierry Pinheiro Moreira. – Campinas, SP : [s.n.], 2014.

Orientador: Hélio Pedrini.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Visão por computador. 2. Descritores. 3. Reconhecimento de padrões. 4. Processamento de imagens. I. Pedrini, Hélio,1963-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Reconhecimento de ações humanas em tempo real baseado em figuras de movimento
Palavras-chave em inglês:
Computer vision
Descriptors
Pattern recognition
Image processing
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Hélio Pedrini [Orientador]
Silvio Jamil Ferzoli Guimarães
David Menotti Gomes
Data de defesa: 25-04-2014
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Defesa de Dissertação de Mestrado em Ciência da Computação, apresentada pelo(a) Mestrando(a) **Thierry Pinheiro Moreira**, aprovado(a) em **25 de abril de 2014**, pela Banca examinadora composta pelos Professores Doutores:

Prof(a). Dr(a). Silvio Jamil Ferzoli Guimarães
Titular

Prof(a). Dr(a). David Menotti Gomes
Titular

Prof(a). Dr(a). Hélio Pedrini
Presidente

Institute of Computing /*Instituto de Computação*
University of Campinas /*Universidade Estadual de Campinas*

Real-Time Human Action Recognition Based on Motion Shapes

Thierry Pinheiro Moreira[1]

April 25, 2014

Examiner Board/*Banca Examinadora*:

- Prof. Dr. Hélio Pedrini (Supervisor/*Orientador*)
- Prof. Dr. Silvio Jamil Ferzoli Guimarães
  PUC - MG
- Prof. Dr. David Menotti Gomes
  IC - UNICAMP
- Prof. Dr. Marcelo da Silva Hounsell
  DCC - UDESC (Substitute/*Suplente*)
- Prof. Dr. Anderson de Rezende Rocha
  IC - UNICAMP (Substitute/*Suplente*)

[1]Financial support: CAPES scholarship (process 01P45542013) 2012–2014

Abstract

Human action recognition in videos is an expanding area of knowledge. There is a wide range of possible applications, including user interface, surveillance, smart homes and health monitoring. Most of them require real time responses, however, there is a trade-off between processing time and effectiveness of the recognition, where effectiveness comprises accuracy and robustness in a number of situations. Two main contributions are presented in this work. The first one is a method for obtaining relevant motion information from videos, even by making use of poorly extracted foreground, by joining a temporal window of shapes. The second one is a simple and fast descriptor, based on silhouettes or, generically, on motion shapes, that achieves state-of-the-art accuracy in real time. It is built from the relative positions of interest points chosen as extreme points on the motion shapes. The method is evaluated on three public data sets and the experimental results are compared against others from the literature. Some data sets have manually segmented silhouettes available, allowing us to analyze each contribution separately. In all cases, the features are extracted at high frame rates, greater than required to a real time application.

Resumo

Reconhecimento de ações humanas em vídeos é uma área de conhecimento em expansão. Há uma vasta gama de possíveis aplicações, incluindo interface de usuários, vigilância, casas inteligentes e monitoramento de saúde. A maioria delas requer respostas em tempo real. No entanto, há um equilíbrio entre tempo de processamento e eficácia do reconhecimento, sendo que eficácia compreende acurácia e robustez em múltiplas situações. Duas contribuições são apresentadas neste trabalho. A primeira é um método de obtenção de informação relevante de movimento em vídeos, mesmo usando uma subtração de fundo simples, por meio da união de uma janela deslizante de figuras. A segunda é um descritor simples e rápido, baseado em silhuetas ou, genericamente, em figuras de movimento, que alcança o estado da arte na acurácia em tempo real. Ele é construído a partir das posições relativas de pontos de interesse escolhidos como pontos extremos nas figuras de movimento. O método foi avaliado em três bases de dados públicas e os resultados experimentais são comparados com outros da literatura. Algumas bases possuem disponíveis silhuetas segmentadas manualmente, permitindo a análise de cada contribuição separadamente. Em todos os casos, as características foram extraídas em altas taxas de quadros por segundo, mais que o necessário para ser aplicado em tempo real.

Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

BoVW	Bag-of-Visual-Words
BoW	Bag-of-Words
CCR	Correct Classification Rate
CMS	Cumulative Motion Shapes
CNN	Convolutional Neural Network
CoP	Clouds of Points
DTW	Dynamic Time Warping
FPS	Frames Per Second
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradients
IB	Information Bottleneck
K-NN	K-Nearest Neighbors
KTH	Kungliga Tekniska Högskolan (Royal Institute of Technology)
LOAO-CV	Leave-One-Actor-Out Cross Validation
LOO-CV	Leave-One-Out Cross Validation
LOSO-CV	Leave-One-Sequence-Out Cross Validation
LPO-CV	Leave-Part-Out Cross Validation
MAS	Manually Annotated Silhouettes
MKL	Multiple Kernel Learning
MoSIFT	Motion + Scale-Invariant Feature Transform
MSC	Multiple Subsequence Combination
MuHAVi	Multicamera Human Action Video
MuHAVi-MAS	MuHAVi Manually Annotated Sequences
PCA	Principal Component Analysis
PWF	PairWise Features
SEG-CCR	Segment-level CCR
SEQ-CCR	Sequence-level CCR
SIFT	Scale-Invariant Feature Transform
SLA	Sparse Linear Approximation
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine

Chapter 1

Introduction

1.1 Motivation

Human action recognition has a wide range of applications and can be used for tasks such as intelligent surveillance, human-computer interface, smart homes, health monitoring, and augmented reality. Through the images obtained by common cameras, the computer is able to perceive some pre-trained actions that people perform and react to it accordingly.

The development of digital technology made way to the progress in the area of action recognition. Cameras have been developed with smaller physical dimensions, as well as higher resolution and frame rates. Images acquired by cameras have been recorded in larger quantity, due to the increase of storage capacity of the digital media (Sasse 2010). The range of devices capable of quickly processing images has also grown; applications are made not only for traditional computers, but also to smartphones, cars and video games, mobile and consoles.

Current researches in surveillance, in general, focus on the development of intelligent surveillance systems that aim at interpreting human activity, instead of using a passive monitoring system, which is the most commonly employed technology (Jacques Junior et al. 2010). Intelligent systems allow the reduction of the necessity of monitoring operators and can help the analysis of images and videos. Nevertheless, intelligent monitoring systems should be capable of automatically extracting complex information of the observed scene and classify its main events – actions or activities. Automated human action recognition is fundamental in surveillance tasks since human beings are susceptible to failure under stress and repetitive conditions. Moreover, camera recordings are often just stored, without being watched or verified anyhow, except in case of casualties.

Home automation often includes lighting control, temperature control, locking of doors and gates, and fire safety. For example, an intelligent system, capable of identifying actions, may be able to detect if a person is reading under appropriate lighting. It may

be used for improving health or increasing quality of life. For elderly and disabled people, fall detection can produce an alert in case of emergency.

Action recognition in human-computer interface is also in growth. It is increasingly used in many contexts. One growing trend is to operate devices using hand and eye gestures; besides practicality, it is used to interface people with disabilities. Another trend, which has a large money turnover, is to control video games using body movements. These devices, however, have well defined scopes and they often include 3D cameras or infrared sensors.

One of the main challenges related to this problem is the computational time required to process video frames and extract features, which often makes it impossible to apply action recognition in real life situations. The achievement of the required processing speed is strongly related to loss of effectiveness in the classification.

1.2 Objectives and Contributions

Two contributions are given in this work. The first one is a shape-based pose descriptor. It is computed from the borders of silhouettes or, more generically, shapes representing the movement on a scene. It is fast to compute, since it is built by the position of discriminative sampled points, and has low dimensionality, which means that classification machines should work fast with it.

A great disadvantage of using silhouettes for action recognition is their calculation. Very often, foreground detection methods return broken and noisy shapes. This is where the second contribution lies: a strategy for making better use of poorly extracted silhouettes to build the descriptor, called Cumulative Motion Shapes (CMS). Due to the faulty nature of the foreground, the concept of silhouette is extended to shapes, in a generic manner. Broken shape components are reattached to form a meaningful shape, more suitable for description.

The proposed action recognition method is evaluated on three widely used public data sets – Weizmann, KTH and MuHAVi –, which have different actions, with different complexities. Experimental results demonstrates state-of-the-art accuracy in small processing time. The descriptor is applied directly, without aggregating other techniques to the classification, which could improve the quality of the solution. The program runs with frame rates higher than necessary to be real time.

1.3 Organization

The text is organized as follows. Basic concepts for the full understanding of this dissertation are explained in Chapter 2. An overview of recent related work is included in Chapter 3, explaining the main adopted strategies and methods that implement them, and comparing their results. The proposed methodology, including the segmentation, filtering and detection of reference point stages, the construction of the descriptor, and the classification process are explained in Chapter 4. The experimental results obtained by applying the method to three public data sets are presented and discussed in Chapter 5. Chapter 6 concludes the dissertation and includes some directions for future work.

Chapter 2

Basic Concepts

This section introduces some relevant basic concepts for a better comprehension of the subjects discussed in this dissertation. Discussions are made on the parts of video streams, in Section 2.1; definition of action and the difference from activity, in Section 2.2; segmentation of movement, in Section 2.3; tracking of movement, in Section 2.4; metrics for method validation, in Section 2.5; as well as real time applicability on a video processing program, in Section 2.6.

2.1 Video

A digital video is an ordered set of images with same dimensions. Each of these images is called a frame. The frequency at which an imaging device captures or exhibits frames is known as frame rate, or frames per second (FPS). These frames can be grouped into shots, which are contiguous subsets of frames that represent continuous actions in time.

A video scene consists of a sequence of semantically correlated shots (Lin & Zhang 2000). Figure 2.1 illustrates such structure. The location in which the actions happen, with all the objects and actors, is referred to as scenery. Frequently, a video, as a whole or only a relevant part, is called a frame sequence or a video sequence.

2.2 Actions

The terms "actions" and "activities" are not always clearly defined in the literature. "Action" means a simple pattern of human movement – such as walking or taking steps, waving hands and collapsing – and can afterwards be used to infer an "activity", which corresponds to a complex task that involves the identification of several actions, interaction between individuals and interactions with objects on the scene (Turaga et al. 2008).

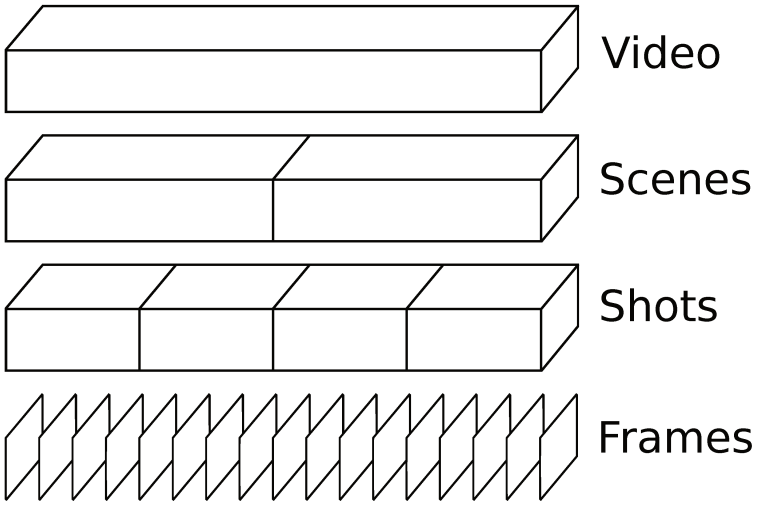


Figure 2.1: Structure of a digital video.

Examples of activities are person tracking, jumping a turnstile or convulsing (Alcântara et al. 2013). The focus of this work is on the domain of actions.

The complexity of the action pattern may change. For example, the system may be trained to understand a step as an action. Thus, walking may consist of a repetition of steps or a single action, periodical and variable in length.

2.3 Movement Segmentation

The goal of movement segmentation is to identify moving objects in the scenery. Subsequent classification process steps strongly depend on this stage. Hu et al. (2004) categorize the movement segmentation into three types: background subtraction, temporal difference, and optical flow.

Background subtraction is a simple and common method. The current video frame is subtracted by a background reference image, pixel-by-pixel. This technique is sensitive to changes in the scenery, being intolerant to changes in luminosity, shadows, camera movement and background. Background modeling can reduce the sensibility of this approach (McKenna et al. 2000, Haritaoglu et al. 2000, Stauffer & Grimson 1999).

Temporal difference makes the pixel-by-pixel subtraction of two or three consecutive video frames. This method is more versatile to dynamic environments, but usually fails to find all the relevant pixels, leaving, for example, gaps inside the moving objects.

Optical flow techniques use vectorial flow characteristics in moving objects. They can be based on gradient, region correspondence, energy or filter phases (Barron et al. 1994). These methods are capable of detecting movement even with camera motion. The methods

are complex and, often, need specialized equipment.

2.4 Movement Tracking

After segmenting a moving object in a scene, a correspondence with moving objects in previous frames is performed to identify them as a single entity and apply classification methods correspondingly. This is important in sequences where there are multiple moving objects.

Hu et al. (2004) characterize the tracking methods in four categories: based on regions, active contour, characteristics, and models. These methods are briefly described as follows.

Methods in the first category use regional characteristics, which many times subdivide objects. In McKenna et al. (2000), for example, people and, later, groups of people are tracked by grouping regions through geometric characteristics. These methods are ineffective when there is occlusion and return little information about the movement.

Active contour model tracks objects using shape representations, which are updated in each video frame. These algorithms describe objects in a simpler and effective way, reducing, therefore, computational cost. However, these techniques return only object contour information, such that it is difficult to obtain their pose and orientation. Another problem is the high sensibility to tracking initialization.

Characteristic-based methods recognize and track objects by extracting elements that describe them and compare between frames. These algorithms can be used in real time applications and can track multiple objects simultaneously. However, they present serious disadvantages: low object recognition rate due to perspective distortion, low tolerance to obstructions, and difficulty in obtaining pose and orientation information.

The methods in the forth category use models obtained by previous information about the shape and flexibility of the tracked objects, where modeling tools or computer vision techniques are used to obtain this knowledge. These algorithms have some advantages: they usually are robust due to employ prior information about the objects; they are tolerant to occlusions; other types of information can be combined to shape, such as structure and articulation; it is easy to estimate the three-dimensional position of the objects through camera calibration; they are robust to high variation in the orientation of the objects.

A widely used tracking technique, independently of the aforementioned classification, is the Kalman filter (Kalman 1960). It is a recursive method that operates in linear data filtering. The Kalman filter estimates the next stage on a discrete temporization process from the current stage and previously collected information. The technique stores in memory only the current stage and some previous information, so that it is not necessary

to store previous frames for a video application. These characteristics make such technique very useful for movement tracking.

2.5 Validation

Most of the methods available in the literature validate the results through leave-one-out cross validation (LOO-CV) scheme. This cross validation separates one instance for testing and use all the others for training, doing it for every instance of the data set.

The correct classification rate (CCR) is computed by Equation 2.1. Another less often used option is the leave-part-out cross validation (LPO-CV) scheme, in which the data set is divided into n parts. For each part, P_i, it is separated from the rest to be used for testing, and all the others are used for training. A CCR is computed for each part, similarly to the previous method, and the overall CCR is calculated through their mean, as shown in Equations 2.2 and 2.3.

$$\text{CCR} = \frac{\text{correct classified instances}}{\text{total number of instances}} \tag{2.1}$$

$$\text{CCR}_{P_i} = \frac{\text{correct classified instances in the part } i}{\text{total number of instances in the part } i} \tag{2.2}$$

$$\text{CCR} = \frac{1}{n}\sum_{i=1}^{n}\text{CCR}_{P_i} \tag{2.3}$$

A particular case of LPO-CV is when the parts are chosen as the group of sequences of each actor. This is called leave-one-actor-out cross validation (LOAO-CV) scheme.

Commonly, action recognition methods need only a small shot of the video or even a single frame, therefore allowing to extract several instances from each video sequence. In this case, instances of the same video may not be tested against each other. Therefore, whenever an instance is tested, all the instances from the same video must be excluded from the training set. This is frequently referred to as leave-one-sequence-out cross validation (LOSO-CV) scheme. The results of this validation are called segment-level correct classification rate (SEG-CCR). In this context, the results of the original type of cross validation are called sequence-level correct classification rate (SEQ-CCR).

For comparability with the literature, all experiments related in this work use the leave-one-out cross validation scheme.

2.6 Real Time Applicability

Specifically, in the context of videos, an application can be considered to work in real time if the output frame rate is at least as high as the input frame rate. Video frame rates may vary: NTSC television's is 23.976 FPS, PAL television's is 25 FPS, and movies' is 24 FPS.

In practice, the minimum frame rate depends on the application. Frame rendering to exhibit a film or a show on a screen requires the aforementioned frequencies. There is no standard for user interfacing, however, users may be very sensitive to delay, so they may be unhappy with low rates. Analysis of a surveillance scene, on the other hand, may not need to be so fast – people will not move fast enough to pass unseen at a frame rate as low as, for example, 10 FPS, which corresponds to 0.1 seconds per frame.

Processing time depends on the computer employed to run the program and the size of the input data. Hence, in this dissertation, the size of the data and the specifications of the computer where the method runs are described in the discussion of the results – Chapter 5. The bound used as reference is 25 FPS (40 milliseconds per frame), as it matches the frame rate of all the three data sets used for the tests and is an upper bound for most video standards.

Chapter 3

Related Work

There are several strategies for addressing the action recognition problem. Each approach has advantages and drawbacks, or it is better suitable for a specific type of application. In this chapter, the described approaches are classified into three categories: appearance-based, in Section 3.1, shape-based, in Section 3.2, and other approaches, in Section 3.3. Additionally, an accuracy analysis of the described methods is included in Section 3.4.

3.1 Appearance-Based Methods

Methods described in this section use appearance information for action recognition. Often, temporal or spatial information are added to the descriptor in order to enhance effectiveness. They work by extracting local information around a set of spatio-temporal interest points (STIP), commonly representing corners in the 3D motion volume. Descriptors are usually constructed by extracting cuboids, which are the small volumes around the STIPs – their neighborhood.

Methods such as Laptev's 3D extension of Harris operator (Laptev 2005), Dollár's method (Dollár et al. 2005), Scale-Invariant Feature Transform (SIFT) (Lowe 1999), and Speeded-Up Robust Features (SURF) (Herbert et al. 2008) perform both stages. The next step of these approaches is clustering the descriptors in appearance classes, or vocabularies, and building histograms, usually called Bag-of-Words (BoW) or Bag-of-Visual-Words (BoVW). The most commonly used clustering algorithm is K-means (Hartigan & Wong 1979).

In the work by Ryoo & Aggarwal (2009), STIPs are extracted by using the method of Dollár et al. (2005) and clustered into a dictionary. To include geometric information, pairwise spatio-temporal relationships are formed, such as *near*, *far*, *before*, *after*, *during*. After that, two three-dimensional histograms are assembled: one with temporal relationships, and the other with spatial relationships, where two dimensions correspond

to characteristic groups and one to their relationships. A correspondence kernel measures the similarities between two histograms making an intersection to count how many points the histograms have in common. The system decides whether the testing video contains an activity or not by measuring the similarities between the video and other training videos containing the activities. Next, for each video of the group, the intersection of the temporal relationship histograms is made, and each pair of characteristics of the result votes for the instants of beginning and end of the action.

Three experiments were conducted on two datasets, KTH and a their own data set, that includes interactions. The first experiment is applied to the first dataset – the method achieved 93.8% accuracy with leave-one-out cross validation. The second experiment tests the ability of the system to detect if an action occurs in a small, segmented, shot of video. Fifty segments were randomly chosen from the second data set, containing simple actions, and twenty segments in which no action happens. A receiver operating characteristic (ROC) curve shows the false positive rates. The third experiment tested the action recognition on the second data set, which contains 6 actions and includes interaction between actors. The obtained accuracy was 70.8%, but the result was not compared with other works of the literature.

In the work by Sun et al. (2009), local and holistic descriptors are joined before clustering and classification. Two local descriptors were used: SIFT 2D in the frame differences and SIFT 3D, an adaptation of the regular SIFT to a 3D volume. The holistic descriptors were also two: Zernike moments (Khotanzad & Hong 1990) in every frame and Zernike moments in the motion energy images (MEI). The MEI is the image that aggregates difference information in a sequence of frames. Descriptors are concatenated and clustered to create a dictionary.

The experiments were conducted on two public data sets, Weizmann and KTH with a leave-one-out cross validation setup. K-means is used to build a Bag-of-Word representation. Experiments were performed to determine the optimum number of words to each descriptor. SVM (Cortes & Vapnik 1995) classifier was tested with two kernels: polynomial and RBF (radial basis function). Combinations two-by-two of the descriptors were made to evaluate their efficiency, however, in the end, all descriptors were fused. The final accuracy rates are 94% on KTH and 97.8% on Weizmann. No processing time was reported, but each of the descriptors are time-consuming, leading us to believe that the fusion of the four descriptors was very slow.

Ta et al. (2010) developed a method that forms pairwise groupings in spatio-temporal information (PairWise Features - PWF). The interest points and spatio-temporal features are extracted using Dollár's method (Dollár et al. 2005). Two interest points are united in a PWF if they are close spatially and temporally. Local appearance information of both STIPs are concatenated to form the PWF appearance descriptor, and a geometric vector

from the first point to the second one forms the geometrical descriptor. The clustering and BoW processes is applied to each descriptor, forming two histograms per action sequence. The two histograms are combined by concatenation into one feature vector. SVM is used to classify the actions.

For testing purposes, the authors established arbitrary spatial and temporal threshold distances to form the PWFs. Arbitrary dictionary sizes were also established for each data set. Because the method is very time-consuming, the number of features is limited to 1500; this is done by measuring the product of the response functions of both STIPs that formed each PWF. The accuracy rate obtained for the KTH data set was 93.0%, and, for Weizmann, 94.5%. Even limiting the quantity of PWFs, the method was very time consuming.

Wu et al. (2010) developed a hierarchical action recognition framework. The first level recognizes poses, or coarse level actions, such as standing, sitting and lying. This is done mainly by using the aspect of the bounding box by three-dimensional estimation. Actions are refined by combining the BoW strategy to the location in which the action happens; for example, the action reading may happen in the living room or in the study room, and is done while sitting. Their definition for the possible combinations are perhaps too simple (it does not consider that a person can read while lying). Finally, three strategies are proposed to allow multiview: 1) best view - chooses the view with more STIPs in the time interval; 2) combined view - concatenates the histograms of all views in a single descriptor; and 3) mixed view - a single BoW is acquired by using information of all views together.

Four experiments are conducted on a multiview data set presented in the paper. Three correspond to each of the multiview strategies. Best view achieved the best accuracy, while mixed view, the worst. The last experiment trains actions in one camera and tests with another in order to evaluate the transferability of the strategies. Since combined view, by definition, cannot be applied to a single camera, it was excluded from this test. The accuracy of the mixed view strategy was much greater than best view, showing that it is more generic, allowing the training from one environment to be used in the other.

Bregonzio et al. (2012) used the global distribution information of interest points to acquire geometrical information of the action, where actions are represented as clouds of interest points (CoP) accumulated at different temporal scales. A new STIP detection method is also proposed. Interest points are accumulated over time at different time scales to form multiple clouds. Features are computed from the clouds – which generate geometrical description – and are fused to the, more conventional, appearance descriptor based on BoW by using multiple kernel learning (MKL).

Tests were made using LOAO-CV on Weizmann and KTH datasets. First, the proposed CoP representation is tested against BoW: the proposed method itself enhanced the

accuracy in 6-7%. Second, the combination of the descriptors is tested by using concatenation and MKL. Concatenation achieved worse results than just the CoP descriptors, while MKL had results 1-2% better than the single descriptor. Third, the interest point detection was tested against (Laptev 2005) and (Dollár et al. 2005). The proposed interest point detection outperformed both with the proposed representation. Other tests were performed to tune the parameters, but the authors did not compare the method with the state-of-the-art. The average time for feature extraction on Weizmann was 12.8s (more than five times the average length of the videos) and 64.92s for KTH (more than three times the average length of the videos).

Zhang & Tao (2012) used a different learning approach. Described in (Wiskott & Sejnowski 2002), slow feature analysis (SFA) produces slow varying outputs from fast varying inputs. According to the authors, a similar process happens in the human brain. Cuboids are extracted from randomly chosen points over movement silhouette. To improve temporal information on cuboids, they are reformatted, so that they are transformed in a sequence of three frames, as in a sliding window. Since such transformation increases dimensionality, PCA (Pearson 1901) was then applied.

In addition to the original unsupervised SFA, three other models were proposed: supervised SFA, which learns the features for each class separately, creating misleading information because of similar cuboids in different classes; discriminative SFA, in which discriminative dictionaries and functions are created to make intraclass information vary slowly and interclass information vary quickly; and spatial discriminative SFA, that considers the spatial location of the cuboids to infer about the body parts. Accumulated squared derivatives are computed from the outputs to measure the fitting degree from cuboids to the slow feature functions. Linear multi-class SVM is used for the classification.

A number of conditions were experimented. The first experiment served to determine how SFA would be applied and which kind of data would be used as input – some of the conditions tested were whether local or holistic features would be used, how to choose STIPs and how to reduce dimensionality. It determined some specificities described above. The second served to tune the slow feature functions and to determine the effectiveness of each aforementioned variations of SFA were more meaningful. Discriminative and spatial discriminative SFA seem to carry finer information. The other four experiments validated the method on four public data sets. The method achieved state-of-the-art accuracy on KTH, low CCR on Weizmann, overcame all the works presented by the authors on UT-Interaction. Authors compared results only between their method and BoW.

Onofri & Soda (2012) claims that using a whole video to recognize an action is not as effective as using small non overlapping portions of the video. The work extracts features of the portions by using the MoSIFT method (Chen & Hauptmann 2009) – which selects

only the SIFT points of portions with high optical flow values, according to a threshold – and constructs a BoVW; afterwards, information bottleneck (IB) (Tishby et al. 1999) is applied to reduce dimensionality and further enhance CCR. Classification is done and multiple subsequence combination (MSC) is applied by building a matrix containing the probabilities of each class for each subsequence. Then, four criteria are applied on the probabilities of each class; average, maximum, minimum and product; and the class with the best output is selected.

The method was tested against using the whole sequence for the classification; MSC obtained improvements of 1.2%, 1.4% and 9.1% on KTH, UCF Sport and YouTube data sets, respectively. The quantity and length of the subsequences were also tuned, concluding that, for subsequences of size n and video of size N, the best number of subsequences to extract from each video is $\lfloor N/n \rfloor$. The optimum value of n varies according to the set tested; smaller values of n can reduce CCR, however, can speed up the method. Thus, parameters can be tuned in terms of speed or accuracy.

3.2 Shape-Based Methods

The nature of the shapes can be very distinctive, for instance, human silhouettes, movement shapes, relative positions of body parts or pose estimation (through models or even appearance descriptors, as long as they are used for shape description). Such approaches frequently use movement segmentation to obtain the silhouette or to narrow down other searches. Common ways to describe a shape are through functions such as 2D Radon transform (Deans 2007) and shape signatures extracted from a centroid, frequently generated by applying distance functions to each border point in a radial scheme or by dividing the polar space in bins. Thus, these methods are often fragile to the conditions that hinder the motion segmentation, such as luminosity variations, and are not robust to occlusions. On the other hand, they usually result in simpler, yet meaningful, descriptors, which may allow faster execution.

The work described in Singh et al. (2010) uses silhouettes for action recognition. A minimum size to fit all silhouettes over time is computed, a new space-time volume is built with the computed size and the time span of the original sequence, and the figures are resized so that the bounding boxes correspond to the whole image, on the new volume. The frames are divided into a grid, generating subvolumes. A mean-power spectrum is calculated from the frequency spectrum of each pixel in the bins; this way, each subvolume has its own descriptor vector. All vectors are concatenated to build a final descriptor. Their work introduced the MuHAVi (Multiview Human Action Videos) data set and created the baseline for future tests on it. Three cross reference methods were executed: leave-one-out, leave-one-actor-out and leave-one-camera-out.

Frequently, human beings are able to identify an action without seeing the sequence of movement; just the pose often reveals the action a person is doing. The approach in Raja et al. (2011) tries to build a method with such ability, manually annotating a small set of frames, and leaving the program to annotate the rest. In each frame, the position of each body part – head, hands and feet – is located with respect to a bounding box. A description of the pose is made by maximizing an energy function, which takes into consideration pose elements and the action performed. A graph is constructed by linking each labeled image to its nearest unlabeled neighbors based on a distance measure. Then, unlabeled images are linked to their nearest neighbors, labeled or unlabeled. Images are then labeled by optimizing the global energy of the graph.

Experiments were conducted on KTH data set. Since the method classifies and labels single frames, the accuracy rates are computed by the number of correctly classified frames divided by the total number of frames, instead of using the number of videos. Four experiments were conducted, gradually inserting parts of the method into the testing. The first three experiments produced low recognition rates, and the last one 86.6% accuracy, which is still low, compared to the state-of-the-art. The method is only partially supervised and some key poses are annotated, which means that training of a new set is very laboring.

Hsieh et al. (2011) presented a silhouette-based method, which represents the shape by histograms. The silhouette is extracted by adaptive background subtraction and mapped into three polar coordinate systems. The first circle includes the whole silhouette, the second, only the top part, which includes arms and head, and the third, only the bottom part, which includes legs. The polar systems are partitioned into several bins, both in the radii and angle coordinates. Silhouette histograms are computed by counting the number of pixels in each bin. The histograms are then concatenated to build the final descriptor of the pose.

For the evaluation of the method, the Weizmann data set videos are divided into several shots of 10 frames each, and with 5 overlapping frames in adjacent shots. The 93 videos of the data set are divided into a total of 961 shots. Leave-one-out cross validation is used over the segmented shots and nearest neighbor classifier (Cover & Hart 1967) is used. To improve both computational efficiency and accuracy, PCA is applied to the samples before testing. The final achieved recognition rate is 98.3%.

Cheema et al. (2011) developed a method that uses weighted key poses to recognize actions in videos. Pose representation is obtained by a normalized distance function over the sampled contour points. Key poses are computed for each action by K-means clustering, and weights are assigned to each one according to its ambiguity, by counting its occurrence in other classes. In case of a sequence with multiple frames, a weighted voting scheme is used. In case of a single image, a simple key pose matching is done. Experimental results on MuHAVi-MAS and Weizmann data sets showed lower accuracy

compared to the state-of-the-art, however, the method runs in real time.

The work of Karthikeyan et al. (2011) describes silhouettes by 2D Radon transform and its velocity. For each signature, eigen mode and multiset partial least squares mode (to make the system multiview) are computed, resulting in four vectors of 180 dimensions for each camera view. They are concatenated to form the final description. Probabilistic subspace similarity learning (Moghaddam 2002) was adopted, aiming at performing intra-class and inter-class learning. Experiments were conducted with LOAO-CV on the original MuHAVi data set. The achieved accuracy is one of the best in the literature, however, no processing time was reported.

Chaaraoui et al. (2013) developed a silhouette-based method that works in real time. It is based on the principle that a few poses over time are enough to identify actions. A representation of the pose in a given frame is given by a normalized distance signal from all the pixels in the border of the shape to a centroid, calculated as the center of mass of the shape. All the poses are clustered, and key poses are defined by the centers of the clusters. To make the method temporal invariant, Dynamic Time Warping (DTW) is used as distance metric, and the classification is done by nearest-neighbor classifier. To enable multiview, descriptors from multiple cameras are concatenated and treated as one; however, a simple concatenation makes the method work only for the actors are always facing the same direction with respect to the cameras, fact that is not explained by the authors.

Three data sets were used in the experiments. The first, Weizmann, has manually annotated silhouettes available, which were used on the tests. The method ran at 70 FPS, and the accuracy rate on this set was lower than many other works, as expected for a real time application. On the second, MuHAVi, the easier, manually annotated, subsets were used. The method ran at 45 FPS and produced state-of-the-art accuracy. On the last data set, IXMAS, background subtraction was used to obtain the silhouettes. This is a multiview set, in which the actors were free to choose the direction they face in the action. Although the actors did not explain the aforementioned direction problem, the method performed well, achieving good accuracy at 26 FPS.

The work described by Guo et al. (2013) uses shape information to address the problem of action recognition. An empirical estimate of the covariance matrix is computed over the features extracted from a video sample. The log-covariance matrix is calculated by reconstructing the matrix using the logarithms of its eigenvalues. Two classification approaches are considered; the first one is a nearest-neighbor classification using two Riemannian matrix distance metrics; the other one uses sparse linear approximation (SLA) applied to log-covariance matrices using the locations of large non-zero coefficients in the sparse linear approximation to determine the label of the testing sample.

Two strategies are adopted to obtain the feature vectors to be applied in the aforemen-

tioned covariance framework; one is based on silhouettes, whereas the other on optical flow. The first is called silhouette tunnel shapes. They are constructed by the three-dimensional concatenation of the silhouettes in each frame. Then, an overcomplete set of descriptors is built by computing a set of distances from each point of the tunnel to its boundaries in various directions, resulting in 13-dimensional features. The second is essentially similar: optical flow is computed on the video and, for each pixel belonging to a moving object, a set of optical flow based descriptors is computed, including time derivatives and physical measurements, resulting in 12-dimensional features.

The method was evaluated on four data sets: Weizmann, KTH, UT-Tower and YouTube. The tests were conducted using NN and SLA for classification, combined with the feature extraction options – silhouette, available for Weizmann and UT-Tower, and optical flow, in all data sets – with a total of two or four tests on each data set. Each sequence is divided into shots of 8 and 20 frames, with 4 overlapping frames in adjacent shots. On these shots, LOO-CV is used to obtain the SEG-CCR, a voting scheme was built to obtain a classification of the entire video sequence; LPO-CV is also used to be able to compare with other works. The tests showed superior performance of the SLA classifier in most of the tests. The silhouette approach was superior to optical flow by 7-10% on Weizmann data set and 10% on UT-Tower.

Chaaraoui & Flórez-Revuelta (2013) developed a feature subset selection method, which separates the relevant parts of the feature vector and excludes subsets that add redundancy or noise to the feature. The descriptor is built by dividing the polar space into radial bins and summarizing the points of each one. The feature vector is formed by the summarization value of each bin. Classification is done by clustering the poses with K-means, obtaining the key poses. Each video is represented by the sequences of key poses and the comparison between videos is done with DTW. A genetic algorithm is employed to determine which of the bins will be used in the classification; the individuals are binary vectors indicating if each bin will be used and the fitness value is the success rate of the classification using the selected bins.

The method was tested on MuHAVi dataset against the same classification method, but without bin summarization and subset feature selection. Bin summarization provided a considerable increase in the accuracy and the full method increased it even more, achieving 100% on MuHAVi-8. Besides the improvements in accuracy, the dimensionality of the data was reduced up to 47%, indicating that noisy and redundant data have been removed. An advantage of the method is that the dimensionality is reduced before feature extraction, differently from PCA that further needs a matrix multiplication. The method is very fast, running at 96 FPS.

3.3 Other Approaches

The work by Wang et al. (2009) implements a real time intelligent surveillance system, robust to horizontal and vertical camera movement (panning and tilting). The movement segmentation is carried out with optical flow. The resulting objects are split in a grid, and a histogram of optical flow is calculated in each block by dividing the directions into eight bins. A set of statistical values is calculated over each bin and the descriptor is built by the concatenation of all values. In addition to these features, shape and trajectory are also employed. All the variables are used separately as weak descriptors. The system learns the action from every frame, so that each frame of the test videos also has a response. The output for the entire video is generated by a voting scheme. The authors also contributed with the CASIA data set of surveillance scenes.

The introduced CASIA data set is split in half to obtain the training and the testing sets; grid size varies from two to five. The best recognition rate was achieved with the highest grid size, 92.5%. For the Weizmann data set, a Gaussian noise was inserted to test against detection noise. Leave-one-out cross validation was used on this set, for the sake of comparison with other works, and 93.3% accuracy rate was achieved. Finally, the system was tested in viewpoint changes and showed robustness in variations of at most 30 degrees. In all cases, the method worked in real time.

Junejo & Aghbari (2012) developed a method that used trajectory of reference points of actors for action recognition. A method called symbolic aggregate approximation (SAX) is introduced to transform trajectory time-series into symbolic representation. This way, distances between trajectories are approximated to the distance between their representations. Velocity, acceleration and curvature information is added to the descriptor to enrich the classification. Nearest-neighbor classifier is applied on the samples.

Experiments were conducted using 13 body joint reference points. The additional descriptors, velocity, acceleration and curvature, are also validated. The results on MoCAP data set showed that the best average CCR is obtained by appending only curvature information to the descriptor, with results comparable to the state-of-the-art. For Weizmann data set, the best setup was obtained only through trajectory information, however, achieving results inferior to the state-of-the-art. Although the accuracy of CCR was not satisfactory, the computational time of the method is low. However, the joint tracking is not completely automated; complete automation of this task may be slow and susceptible to errors.

Ji et al. (2013) developed a method based on deep learning. Convolutional neural networks – which are commonly applied to still images, building high-level descriptors from low-level ones by using several layers – are extended to the three-dimensional space, so that temporal information is not lost, considering movement information in consecutive

frames. Prior to the convolutions, five channels of information are obtained by applying filters that, for each frame, it obtains the gray values of the features, x and y directions of gradient and optical flow. Convolutional neural networks and subsampling filters are then applied alternately on a sequence of seven frames; these operations transform the video volume into a feature vector. Since actions may take more than seven frames to occur, and the usage of more frames may be problematic, global information about the action is passed to the last neural network layer. The features used for the global descriptors are Bag-of-Words constructed from SIFT descriptors, computed over raw gray images and motion edge history images (MEHI) (Yang et al. 2009).

To test the method in the TRECVID real surveillance data set, where there are several people moving in the scenery, a method for detecting people was applied to segment the cubes containing each person. Experiments were conducted by comparing the presented 3D CNN with the original, 2D, CNN and other state-of-the-art works; their work achieved a 7-10% increase in performance over 2D CNN and around 3% over a state-of-the-art approach. To test the method in the KTH data set, the people detection is replaced by background subtraction, since there is only one actor in the scene and it is simpler to apply it. Sixteen random actors were selected for the training, where the other nine were used on the tests. The comparisons performed on the previous data set are not carried out on the second one, but the overall results of the proposed method are lower than the state-of-the-art.

The works by Moghaddam & Piccardi (2010) and Moghaddam & Piccardi (2013) contribute to enhance the classifier, independently of the features extracted. The initialization of training parameters of hidden Markov models (Baum & Petrie 1966) is crucial to find optimal parameters in short processing time. The method itself for obtaining the descriptors is not the main issue, since the focus is on the classifier, which can be trained with data from any kind of descriptor. Moghaddam & Piccardi (2010) measure accuracy by comparing the initialization obtained through the method against random centers and their average. Moghaddam & Piccardi (2013) conducted tests with three types of description: silhouette projection histogram, silhouette sectorial extreme points and STIP-BoW. The authors did not test shape-based methods on the same conditions as appearance-based methods, however, showed that sectorial extreme points achieved better accuracy than projection histogram.

3.4 Summary of State-of-the-Art Results

This section summarizes the results obtained with methods available in the literature. Details related to the data sets will be presented in Chapter 5. Experiments were not reproduced, since most source codes are not available.

Tables 3.1 and 3.2 show the accuracy rates obtained with the methods described in the previous section. The methods are organized chronologically in both tables. Table 3.1 shows results for KTH and Weizmann data sets. The accuracy rates are not always decreasing along the years, since the methods can address other issues than accuracy, such as reduction of processing time, invariance with respect to certain parameters, and reduction of dependencies.

Method	Data Set	
	KTH	Weizmann
Ryoo & Aggarwal (2009)	93.8	-
Sun et al. (2009)	94.0	97.8
Wang et al. (2009)	-	93.3
Ta et al. (2010)	93.0	94.5
Raja et al. (2011)	86.6	-
Hsieh et al. (2011)	-	98.3
Cheema et al. (2011)	-	91.6
Bregonzio et al. (2012)	94.3	96.7
Junejo & Aghbari (2012)	-	88.6
Zhang & Tao (2012)	93.5	93.9
Onofri & Soda (2012)	97.0	-
Chaaraoui et al. (2013)	-	90.3
Ji et al. (2013)	90.2	-
Guo et al. (2013)	98.5	100
Moghaddam & Piccardi (2013)	-	96.8
Alcântara et al. (2013)	-	94.6
Alcântara et al. (2014)	90.1	96.8

Table 3.1: Comparison of correct prediction rates (in percentage) for KTH and Weizmann data sets.

Table 3.2 shows results for the MuHAVi data sets. Additionally to the original set – with long and complex actions – there are two subsets consisting of manually annotated silhouettes for some primitive (simple) actions. The subsets are obviously much easier to classify. The works either use the original set or the segmented subsets. This indicate that the silhouette based methods are not effective on foreground extracted automatically, while appearance based methods cannot extract information using shapes solely.

It is noticeable that the methods that work in real time have achieved smaller correct recognition rates than most of the others. The real-time methods, amongst the ones presented in this chapter are Wang et al. (2009), Cheema et al. (2011), Junejo & Aghbari (2012), Alcântara et al. (2013) and Chaaraoui et al. (2013).

Method	Data Set		
	MuHAVi	MuHAVi8	MuHAVi14
Wu et al. (2010)	69.2[†]	-	-
Singh et al. (2010)	-	82.4	97.8
Moghaddam & Piccardi (2010)	80.4	-	-
Karthikeyan et al. (2011)	88.2	-	-
Cheema et al. (2011)	-	95.6	86.0
Moghaddam & Piccardi (2013)	92.0	-	-
Chaaraoui et al. (2013)	-	97.1	91.2
Chaaraoui & Flórez-Revuelta (2013)	-	100	98.5
Alcântara et al. (2014)	89.1	100	94.1

[†]Experiments conducted by (Karthikeyan et al. 2011).

Table 3.2: Comparison of correct prediction rates (in percentage) for MuHAVi and its manually annotated sub-datasets, MuHAVi14 and MuHAVi8.

Chapter 4

Methodology

This chapter describes the methodology proposed in this work and explains all its parts in details. Figure 4.1 shows a diagram with all five stages of the process, from the data acquisition to a prediction of the action that occurs in the scene. Each section addresses a stage, marked in the figure by letters ranging from (a) to (e).

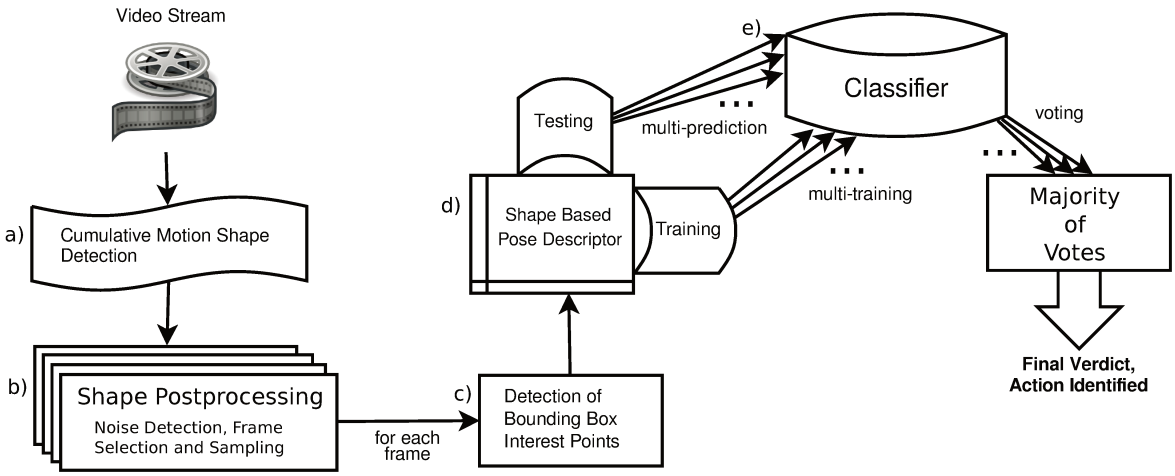


Figure 4.1: Diagram illustrating the main stages of the proposed methodology.

4.1 Cumulative Motion Shape Detection

This module corresponds to Figure 4.1, step (a). A motion shape is the moving part of an action in a given frame of a video sequence. In a perfect scenario, the motion shapes correspond to the silhouettes of the actors. However, extracting good silhouettes from videos is a challenge and there is still no algorithm that extracts them well enough in an

acceptable time. We refer to them as motion shapes since errors are acceptable. Figure 4.2 illustrates this: if a person moves only their arms, the only movement segmented would be the shape of an arm.

(a) Original image (b) Extracted silhouette

Figure 4.2: Example of foreground segmentation in a video in which a person moves only her arms.

Background subtraction described in the work by Kaewtrakulpong & Bowden (2001) is used for the foreground segmentation. This is an adaptive method that learns the background using multiple Gaussian mixture models in each pixel, estimated with expectation-maximization algorithms. Some heuristics are used to detect and exclude shadows. This is not a key point in this work, since its main contribution is the strategy of combining silhouettes and the construction of the descriptor.

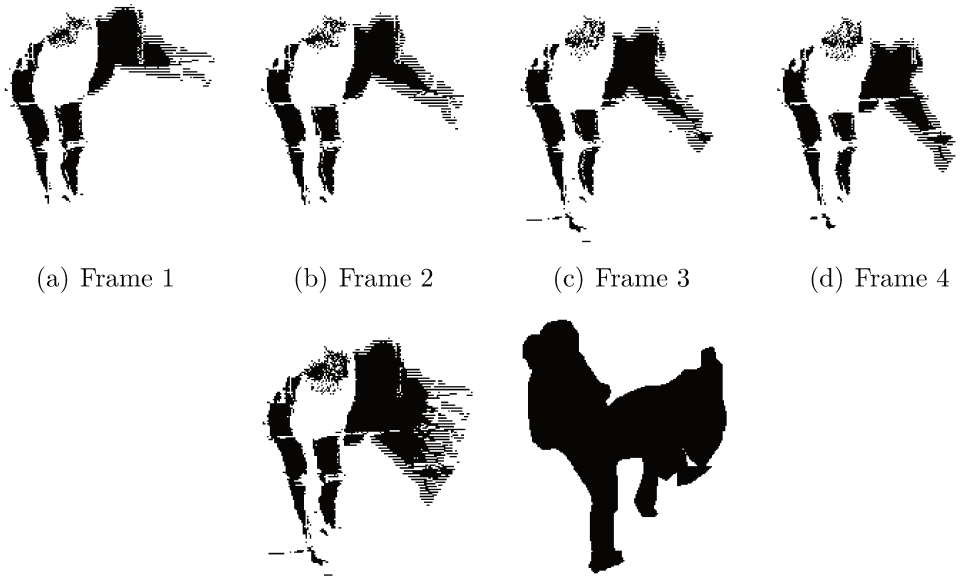
The extracted foreground is subject to noise, missegmentation and disconnections. Therefore, in step (c), morphological operations (Pedrini & Schwartz 2007) are applied to remove it. Firstly, morphological closing is applied, with a 3×3 structuring element, to join fragmented shapes. Then, an area opening is used to remove small, noisy, objects, usually due to small changes in the background and lighting. In experimental tests, any component with a number of pixels smaller than 1/360 of the total image area could be interpreted as noise or useless information, such that these regions are considered background components. Finally, a morphological reconstruction is used to reattach disconnected parts around the remaining components, recovering some fragments subtracted in the previous step. Some frames with outlier values are discarded also in step (c); a frame is considered outlier when the bounding box shows little movement, no movement at all, or when part of the shape is outside the frame.

After extraction of foreground and application of morphological operations, the cumulative motion shapes (CMS) are computed. This is the second contribution, mentioned in Section 1.2. A sliding window is passed on the temporal dimension of the video volume. In each position, the union of all the motion shapes is made, resulting on the CMS that represents the window. The CMS for the k-th frame of a video sequence is given by

Equation 4.1, where n is the size of the sliding window and S_i is the motion shape of the i-th frame.

$$CMS_k = \bigcup_{i=k-n}^{k} S_i \tag{4.1}$$

This process is illustrated in Figure 4.3. From 4.3(a) to 4.3(d), the original poorly extracted foregrounds of four frames are shown – corresponding to a four frames sliding window, on a sample video of *smash object* action of the MuHAVi data set. Figure 4.3(e) shows the CMS constructed from these frames. Figure 4.3(f) shows a CMS from the kick action of the MuHAVi manually annotated subset.



(a) Frame 1 (b) Frame 2 (c) Frame 3 (d) Frame 4

(e) CMS (f) CMS from kick action

Figure 4.3: (a)-(d) Examples of poorly extracted foreground from smash object action; (e) CMS from joining previous images; (f) CMS from kick action. The effect of horizontal lines is due to the interlaced acquisition of the images; it also appears on the original videos. All images are extracted from MuHAVi data set.

Different actions often have common poses. The CMS adds temporal information to poses without raising dimensionality, therefore, neither requiring more processing for pose description nor memory usage. In the case of missegmentation, the CMS sometimes gather the broken portions of movement, setting up a meaningful shape. This is visible in Figures 4.3(e)-4.3(f).

4.2 Shape Postprocessing

This module, corresponding to Figure 4.1, step (b), aims at excluding noisy shapes. Noise can be due to a wide range of phenomena and it is very difficult to deal with it. Some noise types can be filtered in the first step of the process (before or during background subtraction) and some defects caused by the remaining noise are treated with morphological operations, shown in Section 4.1. However, this often results in meaningless information; therefore, among the resulting frames, the faulty images are discarded. A frame is considered faulty when the bounding box shows little movement, no movement at all, or when part of the shape is outside the frame – this may result in partial information. Since the system does not know how much of the shape is excluded, any indication that the shape is partially outside the frame is enough to discard it, as faulty.

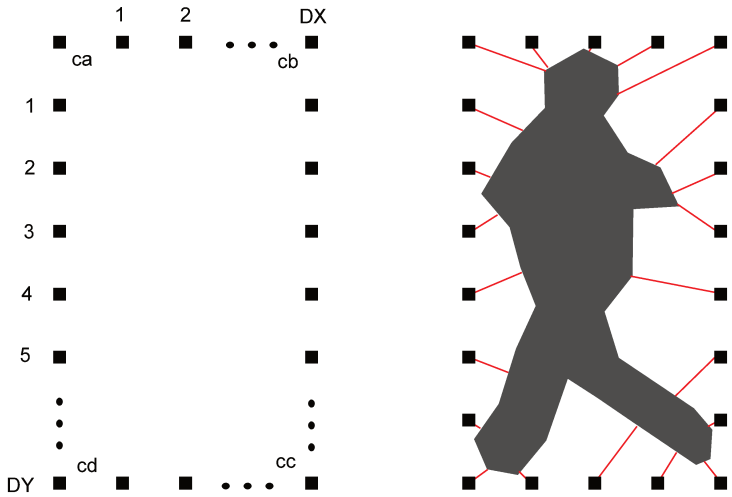
The step starts by computing the bounding box of the motion shape contained in the image. It is defined as the smallest rectangle, with its sides parallel to the axis, that contains the entire shape. Heuristics are used to decide which frames carry meaningful information and which frames are noisy based on the size of the bounding box. The frames discarded are those that:

- there are no bounding boxes: no information available; it happens if there is no one in the scene or if the actors are not moving.
- the bounding boxes are too small: a threshold t is defined; the bounding boxes with width or height smaller than t are discarded.
- the bounding boxes touches the border of the frame: this usually indicates that only a part of the person is visible; partial information would undermine the training.

4.3 Detection of Interest Points

To acquire the interest points (step (c) in Figure 4.1), extreme points are selected on the CMS. In order to find them, key points are equally distributed along the bounding box sides, as represented in Figure 4.4(a). Then, for each key point, the nearest point of the CMS is selected as an interest point, as shown in Figure 4.4(b). The number of key points can be parameterized, however, it is the same over all video streams used in the training and testing processes.

The four corners of the bounding box are denoted as c_a, c_b, c_c and c_d. The k-th subdivision (p_k) between two adjacent corners, c_x and c_y, is represented in Equation 4.2, where D is the number of bounding box subdivisions on the edge between the two corners.



(a) Bounding box subdivision (b) Interest Points

Figure 4.4: Interest points. (a) scheme illustrating how the control points are found in the bounding box; (b) characterization of the interest points by the distances from the CMS to the control points. In (b), we have $(6 + 4) \times 2 = 20$ interest points.

The points are treated as vectors for the operations.

$$p_k = \frac{k.(c_x - c_y)}{D} + c_x \tag{4.2}$$

The bounding box sides are divided by a fixed number of points, which does not need to be the same for all four sides. Eventually, it can be interesting to use a distinct number for horizontal and vertical sides – as done in the experiments reported in Chapter 5. This is because the CMS can have more information disposed in vertical than the horizontal direction.

Euclidean distance is used to find the nearest CMS point from each of the border point, as shown in Figure 4.4(b). The selected contour points of the CMS are the interest points that will be used to build the descriptor.

4.4 Descriptor Construction

The main contribution of this work is the descriptor. For each CMS, (step (d) in Figure 4.1) it is constructed as follows: first, the centroid of the shape is found, as the center of the bounding box; second, a coordinate system is created, using the centroid as origin, and normalized so that the limits of the bounding box are -1 and 1; third, the coordinates of each interest point are obtained, in the new coordinate system; fourth and last, the

descriptor vector is built by concatenating the x and y coordinates; clockwise, starting from top-left. The process is illustrated in Figure 4.5.

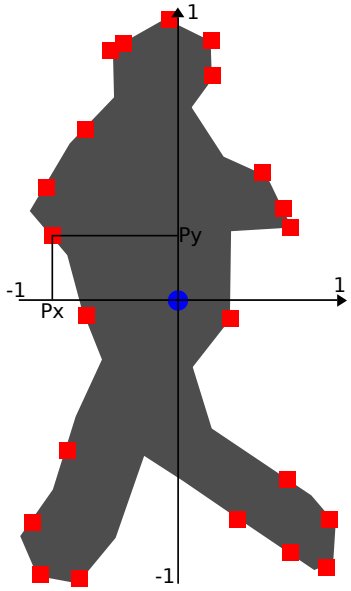


Figure 4.5: Construction of the descriptor from the normalized coordinates of the interest points.

Every frame of the video results in one CMS, except the first frames, before the window is filled, and in the case of discards. Several CMS can result from one action sequence, hence this step ends with multiple descriptors for each input video. Therefore, multiple vectors are passed forward to the classification machine, described in Section 4.5. There is no precedence order between distinct CMS from the same video stream.

Extracting multiple samples from a same sequence helps learning actions starting from any part of its period – for example, a walking action may start with two feet together or after a step has already been taken.

4.5 Classification

Several descriptors are extracted from each video sequence. When the classification machine is trained, each training video sequence contributes with multiple independent instances. This is represented in Figure 4.1, step (e), as multi-training.

Similarly, when a prediction is made, multiple descriptors are computed from the test video sequence. Each one is used to a distinct prediction, and each one contributes as a vote to the final output. The final verdict of the system is the majority in the votes. The creation of the descriptor is the same both for training and testing.

Initially, all the frames can be used by the classifier, but this could cause redundancies. Instead, N equally spaced samples are selected, where N is parameterized. Experiments related to this stage are shown in Chapter 5.

Chapter 5

Experimental Results

In this chapter, the proposed methodology is tested on three human action public data sets. The data sets used are Weizmann (described in Section 5.1), KTH (described in Section 5.2) and MuHAVi (described in Section 5.3).

The parameters of the method are chosen from a grid search scheme and its accuracy is shown through accuracy curves. In each curve, one value is varied, while the other elements of the grid are fixed to the best value. Each data set has its own particularities, which will be discussed in the respective sections. Some sets have manually annotated silhouettes, so the impact and efficiency of an automatic silhouette extraction can be observed. Thus, inferences can be made from the results to determine the effectiveness of the proposed method.

The number of bounding box key points, as described in Section 4.3, was arbitrarily set to 16 in each vertical side of the box, and 8, in each horizontal side. More points were picked in the vertical dimension because the shape of a person standing up is elongated in this direction. This sums up to 48 interest points over the cumulative motion shapes (CMS). Each point results in two dimensions in the shape descriptor – the x and the y dimensions of the normalized coordinates – resulting in a 96 dimensional vector.

Additionally, Principal Component Analysis (PCA) (Jolliffe 2002) is applied to the final descriptor, aiming at speeding up the classifier and enhancing its accuracy. A total of three parameters is required for the system: number of PCA dimensions, number of frames united to build the CMS – referred to as CMS number – and number of samples extracted from the videos. Moreover, there are parameters used for the classification algorithms.

All the experiments were performed by using the proposed multi-training in K-Nearest Neighbor (K-NN) and multi-class Support Vector Machines (SVM) using Radial Basis Function (RBF) kernel. Using K-NN as a classification machine adds one parameter to the list, the number K of neighbors, whereas using SVM as the classification machine

adds two parameters to the list, the soft margin parameter *cost* and the influence distance parameter γ. In all tests, γ was set to $1/8d$, where d is the number of dimensions in the feature vector after applying PCA.

Classification time through K-NN is more sensitive to the number of training samples, whereas, for the (multi-class) SVM, the classification time is more sensitive to the number of trained classes. Hence, the number of training samples extracted from each sequence may vary depending on the classifier.

All the measured times were found by taking the average of 5 runs. The computer used in the experiments was an i7 3.5 GHz with no parallelism mechanism implemented. The feature extraction was coded in C++ programming language with OpenCV library. The classification code was written separately in R package through the machine learning libraries e1071 and kernlab. Time is measured separately for the extraction of features and classification since they are independent processes and can be combined in different ways. Extraction of features involves all the processes starting from reading the video to the construction of the final descriptor – Figures 4.1(a) to (d).

5.1 Weizmann Data Set

Weizmann (Blank et al. 2005) is an action data set consisting of 10 classes: run, walk, skip, jumping-jack (or shortly jack), jump forward on two legs (or jump), jump in place on two legs (or pjump), gallop sideways (or side), wave-two-hands (or wave2), wave one hand (or wave1), and bend. Each action class is performed by 9 actors once, except for one referred to as *Lena*, who performs the actions *skip*, *run* and *walk* twice each, resulting in 93 videos.

The frames were captured at 25 FPS (frames per second), size of 180×144 pixels. All the actions occur in the same static background. Figure 5.1 shows some examples from the set.

The Weizmann data set has a total of 5,701 frames, 228.04 seconds at 25 FPS. The extraction of the features from the frames took a time of 4.85 seconds – average of 1,175.95 FPS, less than 1 millisecond per frame. Best accuracy rates and classification times are shown in Table 5.1.

	SVM	K-NN
Accuracy rate (%)	**97.85**	**97.85**
Classification time (ms)	**3**	**3**

Table 5.1: Accuracy rates (in percentage) and classification time (in milliseconds) for Weizmann data set.

(a) Walk (b) Jack (c) Run

Figure 5.1: Examples extracted from the Weizmann public data set.

Since the data set videos are very short, the background subtractor does not have enough frames to learn the background model. Hence, in this set, the foreground is extracted by using frame difference.

5.1.1 K-NN

The parameters for the best accuracy, shown in Table 5.1, are presented in Table 5.2. Based on these values, accuracies varying *K*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.2(a) to (d), respectively.

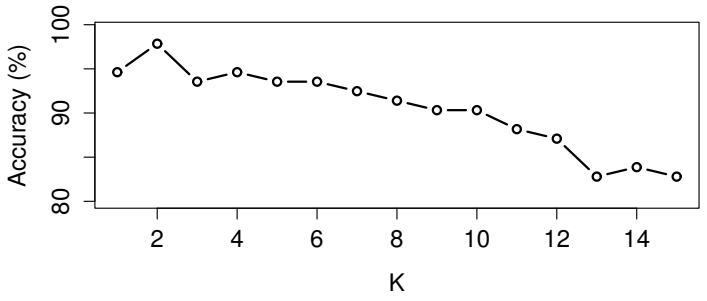
Parameter	Value
K	2
PCA dimensions	20
CMS number	4
Number of samples	15

Table 5.2: Best parameters for Weizmann data set using K-NN classifier.

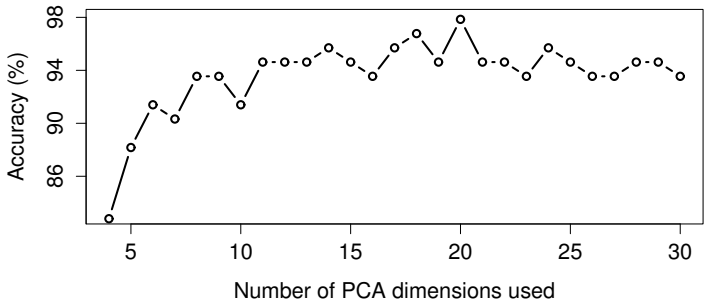
The confusion matrix is given in Table 5.3. It can be seen that all confusions are related to the action *side*, misclassified from similar classes, *jump* and *skip*.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0	0
jump	0	0	0.89	0	0	0.11	0	0	0	0
pjump	0	0	0	1	0	0	0	0	0	0
run	0	0	0	0	1	0	0	0	0	0
side	0	0	0	0	0	1	0	0	0	0
skip	0	0	0	0	0	0.1	0.9	0	0	0
walk	0	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	0	1	0
wave2	0	0	0	0	0	0	0	0	0	1

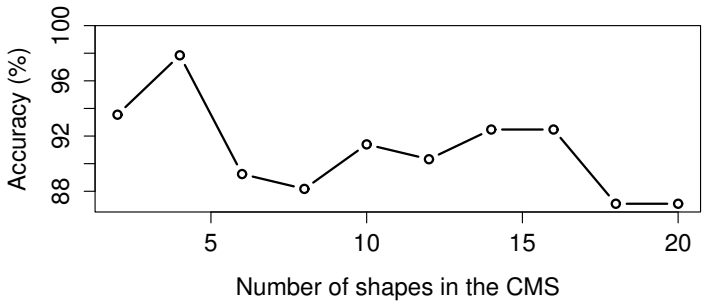
Table 5.3: Confusion matrix of the K-NN results for the Weizmann data set.



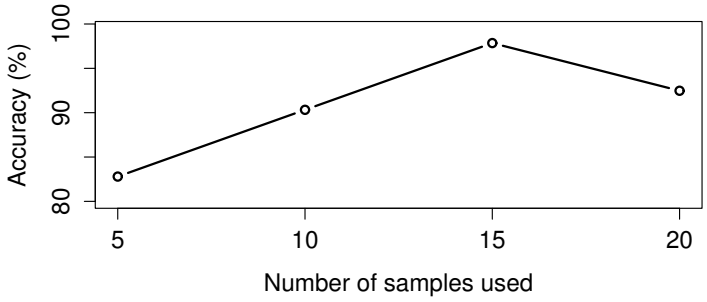
(a)



(b)



(c)



(d)

Figure 5.2: Accuracy curves of Weizmann K-NN tests varying parameter (a) K; (b) PCA dimensions; (c) CMS number; (d) number of samples.

5.1.2 SVM

The parameters for the best accuracy, shown in Table 5.1, are presented in Table 5.4. Based on these values, accuracies varying *cost*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.3(a) to (d), respectively. The confusion matrix is given in Table 5.5.

This is the only case in which the best CMS number is one. This means that the original shapes are used, and that using CMS made the accuracy lower. In all other test setups, this number are higher than one.

Parameter	Value
Cost	1000
PCA dimensions	18
CMS number	1
Number of samples	19

Table 5.4: Best parameters for Weizmann data set using SVM classifier.

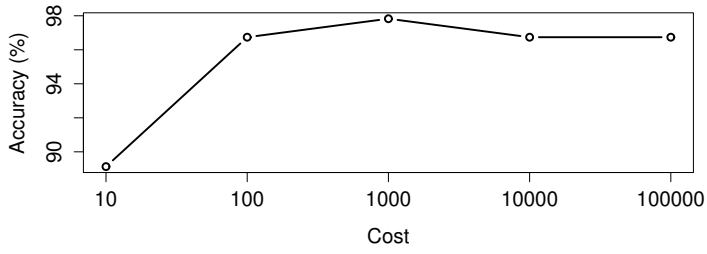
	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0	0
jump	0	0	1	0	0	0	0	0	0	0
pjump	0	0	0	0.89	0	0.11	0	0	0	0
run	0	0	0	0	1	0	0	0	0	0
side	0	0	0	0	0	1	0	0	0	0
skip	0	0	0	0	0.1	0	0.9	0	0	0
walk	0	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	0	1	0
wave2	0	0	0	0	0	0	0	0	0	1

Table 5.5: Confusion matrix of the SVM results for the Weizmann data set.

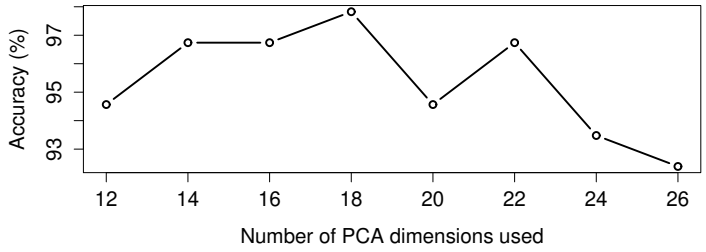
5.1.3 Manually Annotated Silhouettes

The foreground masks are also available for all the videos, corresponding to manually annotated silhouettes (MAS). Figure 5.4 shows some examples. Experiments were made using the masks as the shapes, skipping the first two steps of the methodology – Figures 4.1(a) and (b).

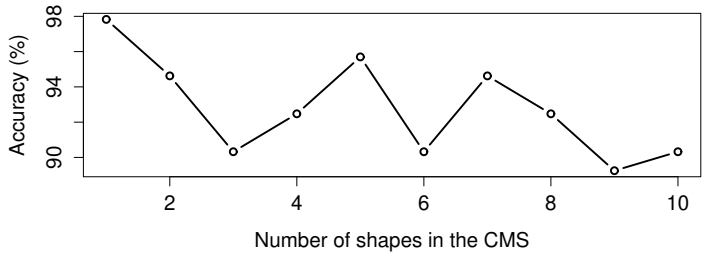
This experiment serves to evaluate the effectiveness of the skipped steps. The masks represent the entire silhouette of the human actor, therefore, there are differences from



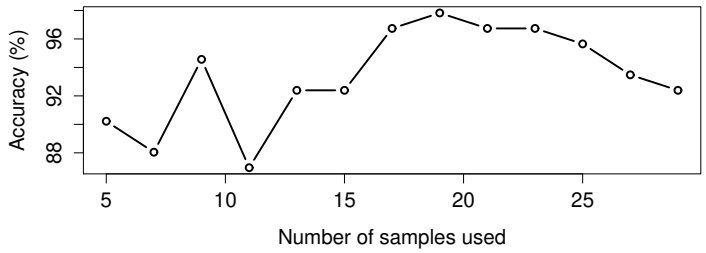
(a)



(b)



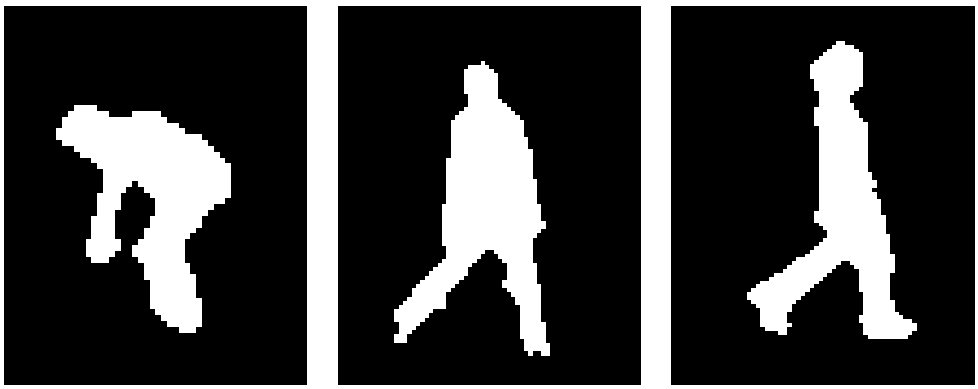
(c)



(d)

Figure 5.3: Accuracy curves of Weizmann SVM tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.

the shapes extracted by the method described in this dissertation – if the person moves only his/her arms, the segmentation process extracts the shape of the moving arms solely, while in the manual segmentation, it would be the entire person.



(a) Bend (b) Side (c) Walk

Figure 5.4: Examples extracted from the Weizmann MAS public data set.

K-NN

The results for the manually segmented silhouettes were as good as the results for the original set. This indicates that the poor quality of the automatic foreground segmentation do not worsen the results. The best result achieved 97.85% accuracy, just the same as the best result for the original set.

Table 5.6 shows the used parameters – very similar to Table 5.2, for the original set. Based on these values, accuracies varying K, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.5(a) to (d), respectively.

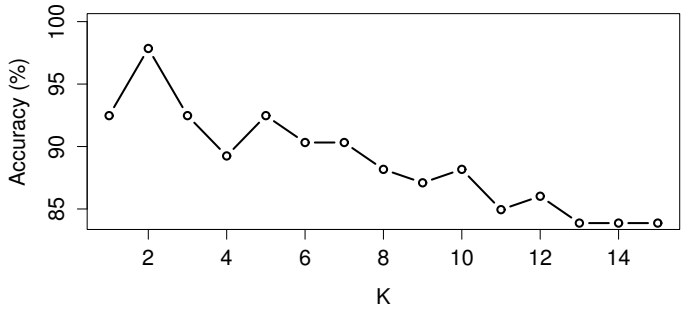
Parameter	Value
K	2
PCA dimensions	12
CMS number	4
Number of samples	17

Table 5.6: Best parameters for Weizmann MAS data set using K-NN classifier.

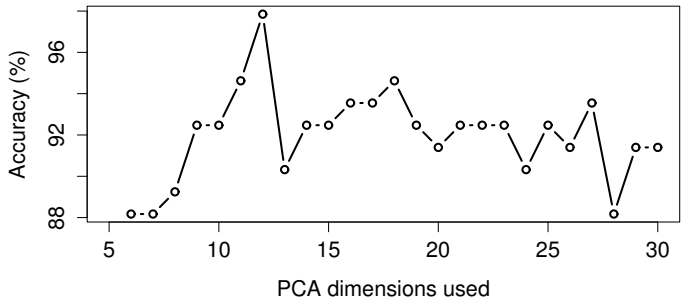
Table 5.7 shows the confusion matrix. Two misclassifications occurred: the first was between *pjump* and *jack*; both involve jumping in place. The second was between *waving one arm* and *waving two arms*, which are very similar actions.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0	0
jump	0	0	1	0	0	0	0	0	0	0
pjump	0	0.11	0	0.89	0	0	0	0	0	0
run	0	0	0	0	1	0	0	0	0	0
side	0	0	0	0	0	1	0	0	0	0
skip	0	0	0	0	0	0	1	0	0	0
walk	0	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	0	0.89	0.11
wave2	0	0	0	0	0	0	0	0	0	1

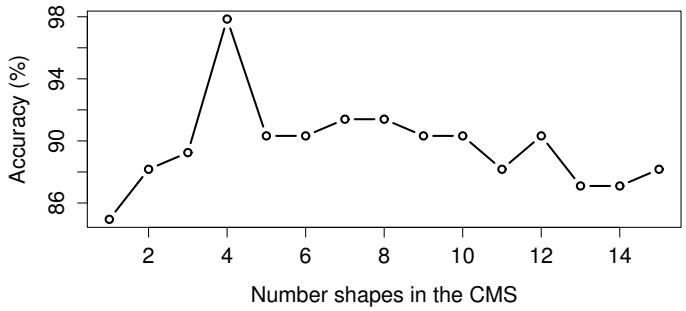
Table 5.7: Confusion matrix of the K-NN results for the manually annotated Weizmann data set.



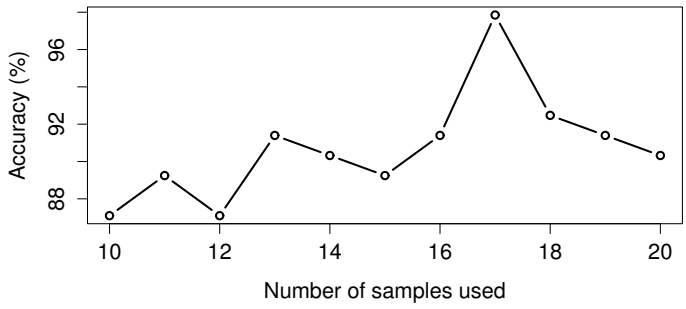
(a)



(b)



(c)



(d)

Figure 5.5: Accuracy curves of Weizmann MAS K-NN tests varying parameter (a) K; (b) PCA dimensions; (c) CMS number; (d) number of samples.

SVM

The results for the manually segmented silhouettes were slightly worse than the results for the original set. The best result achieved 95.70% accuracy, against 97.85% for the original set. This indicates that the effect shown in Figure 4.2 may be beneficial to the description, since the moving parts receive much more emphasis when all the interest points are extracted from them.

Table 5.8 shows the parameters used – very similar to Table 5.4, for the original set. Based on these values, accuracies varying *cost*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.6(a) to (d), respectively.

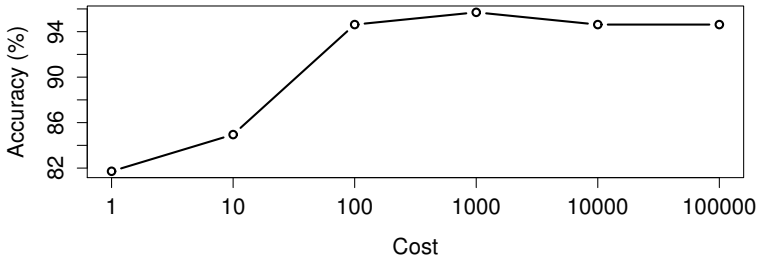
Parameter	Value
Cost	1000
PCA dimensions	17
CMS number	6
Number of samples	16

Table 5.8: Best parameters for Weizmann MAS data set using SVM classifier.

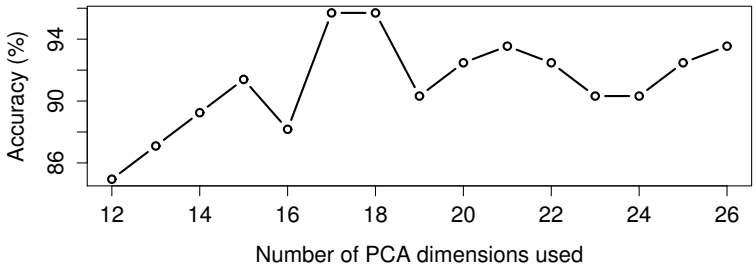
Table 5.9 shows the confusion matrix. Most of the confusion happened between the actions *jump* and *skip*. The actions are similar, as both show the displacement of a person by doing jump-like movements.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	1	0	0	0	0	0	0	0	0	0
jack	0	1	0	0	0	0	0	0	0	0
jump	0	0	0.89	0	0	0	0.11	0	0	0
pjump	0	0	0	1	0	0	0	0	0	0
run	0	0	0	0	1	0	0	0	0	0
side	0	0	0	0	0	0.89	0.11	0	0	0
skip	0	0	0.1	0	0	0	0.9	0	0	0
walk	0	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	0	0.89	0.11
wave2	0	0	0	0	0	0	0	0	0	1

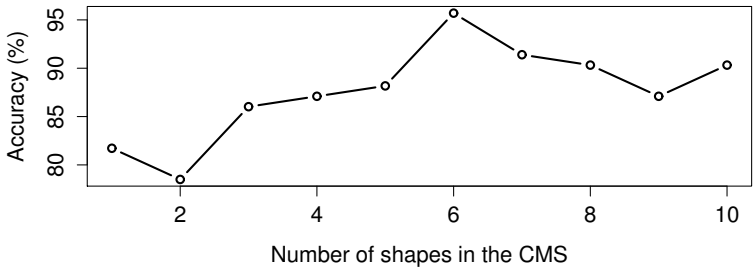
Table 5.9: Confusion matrix of the SVM results for the manually annotated Weizmann data set.



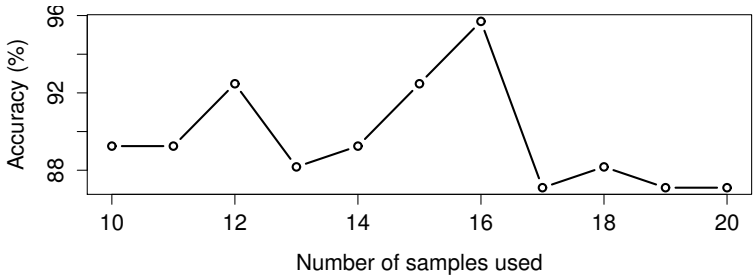
(a)



(b)



(c)



(d)

Figure 5.6: Accuracy curves of Weizmann MAS SVM tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.

5.2 KTH Data Set

KTH (Schuldt et al. 2004) is an action data set consisting of 6 classes: walk, jog, run, boxing, hand wave and hand clap. Each action is performed by 25 actors in 4 different scenes, except for the action *hand clap*, in which one actor performs it in only 3 scenes, resulting in 599 videos.

The frames were captured at 25 FPS, size of 160×120 pixels. Most videos have strong camera movement – zooming, panning and tilting. Camera movements are serious threats for descriptors based on silhouettes or motion shapes, making this data set a challenge for the method. Example frames of the set are shown in Figure 5.7.

(a) Boxing (b) Jog (c) Walk

Figure 5.7: Examples extracted from the KTH public data set.

The KTH data set has a total of 289,715 frames, 11375.32 seconds at 25 FPS. The extraction of the features from the frames took a total time of 1347.38 seconds – average of 215.02 FPS, less than 5 milliseconds per frame. The CMS are constructed using 12 frames and the number of samples extracted for each sequence is 40. Accuracy rates and classification times are shown in Table 5.10.

	SVM	K-NN
Classification time (ms)	43	**23**
Accuracy rate (%)	**90.28**	88.78

Table 5.10: Accuracy rates (in percentage) and classification time (in milliseconds) for KTH data set.

5.2.1 K-NN

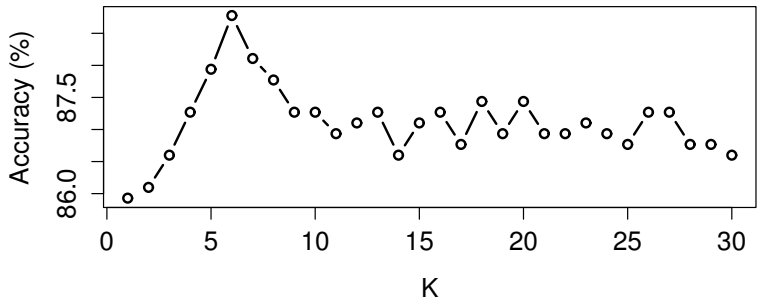
The parameters for the best accuracy, shown in Table 5.10, are presented in Table 5.11. Based on these values, accuracies varying K, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.8(a) to (d), respectively. The confusion matrix is given in Table 5.12.

Parameter	Value
K	6
PCA dimensions	14
CMS number	12
Number of samples	40

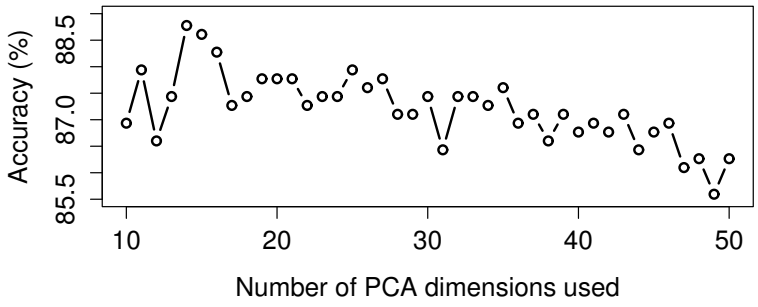
Table 5.11: Optimum parameters for Weizmann data set using K-NN classifier.

	boxing	clap	wave	jog	run	walk
boxing	0.94	0.01	0.02	0	0	0.03
clap	0.02	0.88	0.09	0	0	0.01
wave	0.01	0.02	0.96	0.01	0	0
jog	0	0	0.01	0.84	0.06	0.09
run	0	0.01	0	0.21	0.77	0.01
walk	0.01	0.02	0	0.02	0.1	0.94

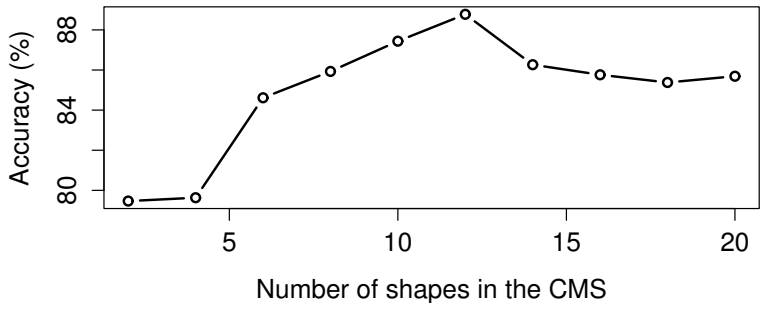
Table 5.12: Confusion matrix of the K-NN results for the KTH data set.



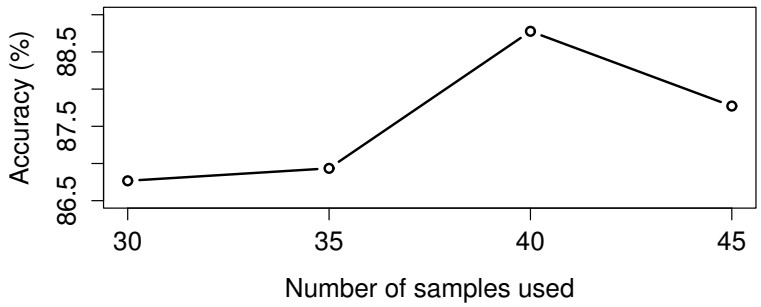
(a)



(b)



(c)



(d)

Figure 5.8: Accuracy curves of KTH K-NN tests varying parameter (a) K; (b) PCA dimensions; (c) CMS number; (d) number of samples.

5.2.2 SVM

The computational time for training and predicting on an SVM machine is much higher than KNN. For a data set the size of KTH, this difference is evidenced; hence, the grid search is made only for two parameters. The CMS number and number of CMS samples used are the same as the best results in KTH: 12 and 40, respectively. The parameters for the best accuracy, shown in Table 5.10, are presented in Table 5.13. Based on these values, accuracies varying *cost* and *PCA dimensions* are shown in Figures 5.9(a) and (b), respectively. The confusion matrix is given in Table 5.14.

Parameter	Value
Cost	100,000
PCA dimensions	30

Table 5.13: Best parameters for KTH data set using SVM classifier.

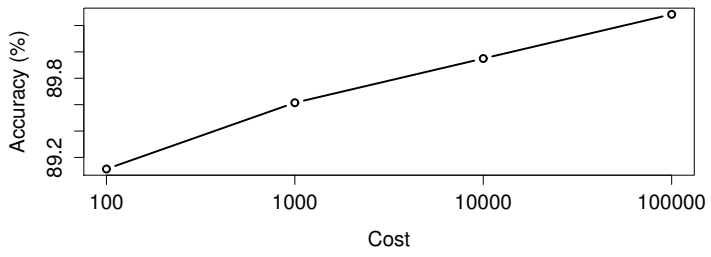
	boxing	clap	wave	jog	run	walk
boxing	0.95	0.02	0.02	0	0	0.01
clap	0.01	0.95	0.04	0	0	0.01
wave	0.01	0.03	0.96	0	0	0
jog	0	0.01	0	0.87	0.09	0.02
run	0	0.04	0.01	0.15	0.79	0.01
walk	0.01	0	0	0.04	0.06	0.89

Table 5.14: Confusion matrix of the SVM results for the KTH data set.

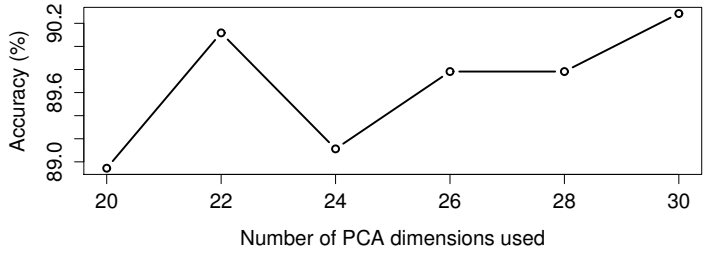
5.3 MuHAVi Data Set

MuHAVi (Singh et al. 2010) (Multicamera Human Action Video Data) is a multiview data set consisting of 17 classes: climb ladder, craw on knees, draw graffiti, drunk walk, jump over fence, jump over gap, kick, look in car, pick up and throw object, pull heavy object, punch, run stop, shot gun collapse, smash object, walk and fall, walk and turn back, and wave arms. Each action is performed by 7 actors, resulting in 119 videos. Many of the videos contain noise movement: people setting up the stage before the action happen and people visibly moving behind the stage – all labeled with the same ground truth of the rest of the video.

This is the most realistic data set, regarding the actions. All the classes are at least related to criminal or suspect behavior (at least in some countries, in the case of *drunk*



(a)



(b)

Figure 5.9: Accuracy curves of KTH SVM tests varying parameter (a) cost; (b) PCA dimensions.

walk). The actions occur in a complex, closed scenario, with 8 cameras surrounding it. Since this work does not focus on multi viewing, only one camera is used, camera 4, which captures the action from the side. The frames were captured at 25 FPS, size of 720×576 pixels. Example frames of the set are shown in Figure 5.10.

The data set has a subset of manually annotated sequences (MuHAVi-MAS), in which the frames are binary images of the silhouette locations. It is divided into 14 primitive actions: collapse left, collapse right, guard to kick, guard to punch, kick right, punch right, run left to right, run right to left, stand up left, stand up right, turn back left, turn back right, walk left to right, and walk right to left.

There are 68 sequences, divided unevenly among its classes. It is usually called MuHAVi14 in the literature. The actions were annotated from cameras 3 and 4; in this case, again, only camera 4 is used.

This subset, however, has some classes that are essentially the same, but vary in direction – for example, *run left to right* and *run right to left*. Another subset was built, rearranging these classes together, forming another subset with 8 classes: collapse, guard, kick right, punch right, run, stand up, turn back, and walk. This rearrangement is usually called MuHAVi8 in the literature.

The MuHAVi has a total of 134,085 frames, 5,368.16 seconds at 25 FPS. The extraction of the features from the frames demanded a total time of 2,850.29 seconds – average of 47.04 FPS, 21.26 milliseconds per frame.

(a) Jump over fence, camera 3

(b) Look in car, camera 4

(c) Wave arms

Figure 5.10: Examples extracted from the MuHAVi public data set.

(a) Kick right

(b) Run left to right

(c) Stand up left

Figure 5.11: Examples extracted from the Manually annotated silhouettes subset of MuHAVi data set.

Since MuHAVi8 and MuHAVi14 have the same video sequences, but in different classes, their feature extraction computational time results are the same. The entire subdata sets have 3,969 frames, 158.76 seconds at 25 FPS. The total extraction of the

features from the frames demanded an average time of 19.54 seconds – average of 203.12 FPS, less than 5 milliseconds per frame. The classification times differ depending on the machine, since the data sets have different numbers of classes.

Accuracy and computational times for classification are shown in Table 5.15.

	Accuracy (%)		Time (ms)	
Data Set	SVM	K-NN	SVM	K-NN
MuHAVi	90.76	**91.60**	259	**6**
MuHAVi14	94.12	**95.59**	48	**1**
MuHAVi8	**100**	**100**	14	**1**

Table 5.15: Accuracy rates (in percentage) and classification time (in milliseconds) for MuHAVi data set.

5.3.1 Complete set

The following results correspond to the MuHAVi data set for both K-NN and SVM classifiers.

K-NN

The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.16. Based on these values, accuracies varying *K*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.12(a) to (d), respectively.

Parameter	Value
K	1
PCA dimensions	24
CMS number	40
Number of samples	55

Table 5.16: Optimum parameters for MuHAVi data set using K-NN classifier.

The confusion matrix is given in Table 5.17, with the action codes specified in Table 5.18.

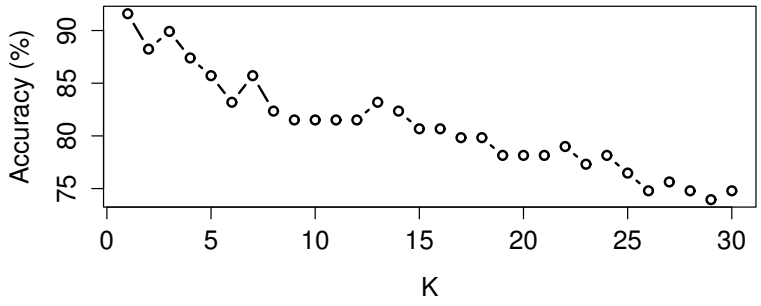
The optimal number of shapes accumulated to build the CMS is higher than the other data sets, since the actions in MuHAVi are more complex and need much more time to take place. Therefore, the actions require many frames to characterize them. For instance, the *climb ladder* action consists of a person walking towards a ladder, climbing it up, down, and then walking away from it.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	.86	0	0	0	0	0	0	0	0	0	.14	0	0
6	0	0	0	0	0	.71	0	0	0	.14	0	0	0	0	0	.14	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	.14	0	.71	0	.14	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	.86	0	0	0	0	0	.14	0
11	0	0	0	0	0	0	.14	0	.29	0	.57	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
17	0	0	0	0	0	0	0	.14	0	0	0	0	0	0	0	0	.86

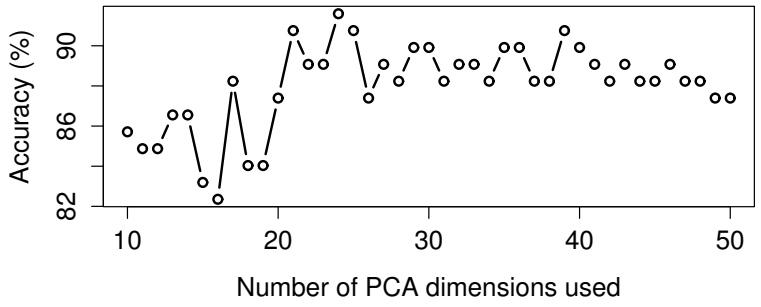
Table 5.17: Confusion matrix of the K-NN results for the MuHAVi data set.

Code	Action class	Code	Action class
1	Climb ladder	10	Pull heavy object
2	Craw on knees	11	Punch
3	Draw graffiti	12	Run stop
4	Drunk walk	13	Shot gun collapse
5	Jump over fence	14	Smash object
6	Jump over gap	15	Walk and fall
7	Kick	16	Walk and turn back
8	Look in car	17	Wave arms
9	Pick up and throw object		

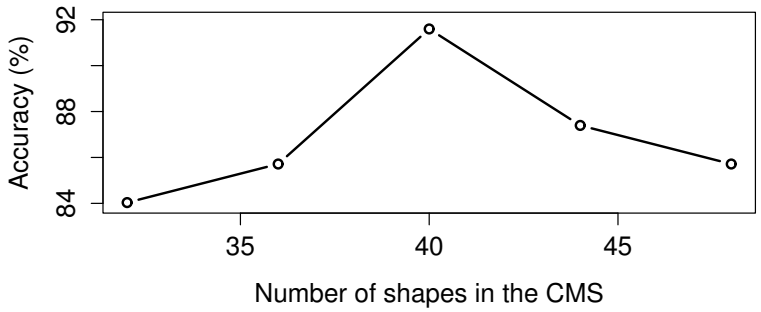
Table 5.18: Action codes for MuHAVi confusion matrices.



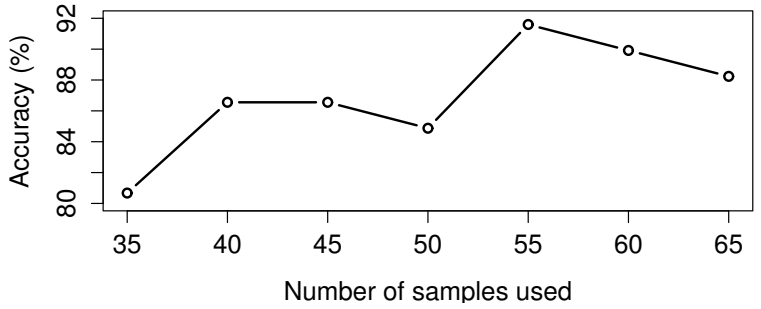
(a)



(b)



(c)



(d)

Figure 5.12: Accuracy curves of MuHAVi K-NN tests varying parameter (a) K; (b) PCA dimensions; (c) CMS number; (d) number of samples.

SVM

The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.19. Based on these values, accuracies varying *cost*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.13(a) to (d), respectively.

Parameter	Value
Cost	100
PCA dimensions	30
CMS number	48
Number of samples	45

Table 5.19: Optimum parameters for MuHAVi data set using SVM classifier.

The confusion matrix is given in Table 5.20, with the action codes specified in Table 5.21.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	.71	0	0	0	0	0	0	0	.14	0	0	0	.14	0
5	0	0	0	0	.86	.14	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	.71	0	0	0	0	0	0	0	0	0	.29	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	0	0	.14	0	0	.14	0	0	.57	0	.14	0	0	0	0	0	0
10	0	0	0	0	0	.14	0	0	0	.86	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	.14	0	.86	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
17	0	0	0	0	0	0	0	0	0	0	0	0	.14	0	0	0	.86

Table 5.20: Confusion matrix of the SVM results for the MuHAVi data set.

5.3.2 MuHAVi14

The following results correspond to the MuHAVi14 data set for both K-NN and SVM classifiers.

Code	Action class	Code	Action class
1	Climb ladder	10	Pull heavy object
2	Craw on knees	11	Punch
3	Draw graffiti	12	Run stop
4	Drunk walk	13	Shot gun collapse
5	Jump over fence	14	Smash object
6	Jump over gap	15	Walk and fall
7	Kick	16	Walk and turn back
8	Look in car	17	Wave arms
9	Pick up and throw object		

Table 5.21: Action codes for MuHAVi confusion matrices.

K-NN

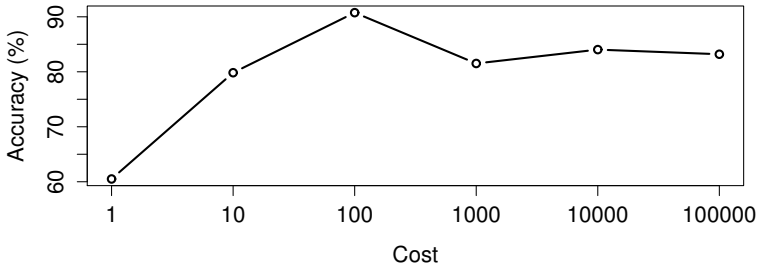
The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.22. Based on these values, accuracies varying K, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.14(a) to (d), respectively.

Parameter	Value
K	3
PCA dimensions	15
CMS number	5
Number of samples	12

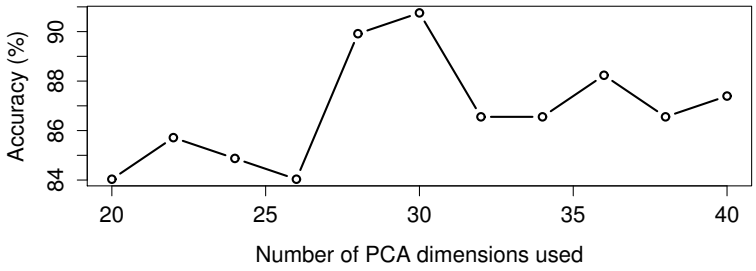
Table 5.22: Optimum parameters for MuHAVi14 data set using K-NN classifier.

The confusion matrix is given in Table 5.23, with the action codes specified in Table 5.24.

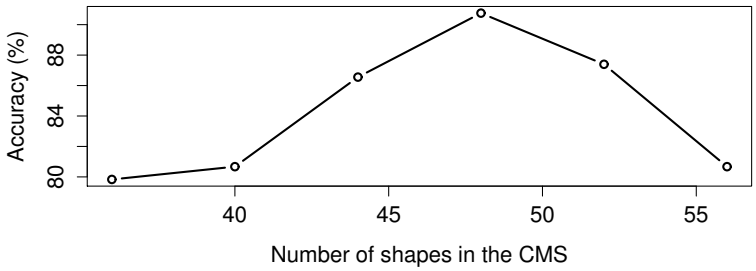
The manually annotated subsets consist of shorter sequences, with simpler actions. Therefore, their actions are better described by CMS constructed by fewer frames. Moreover, since the segmentation was done manually, the CMS loses their restoration importance.



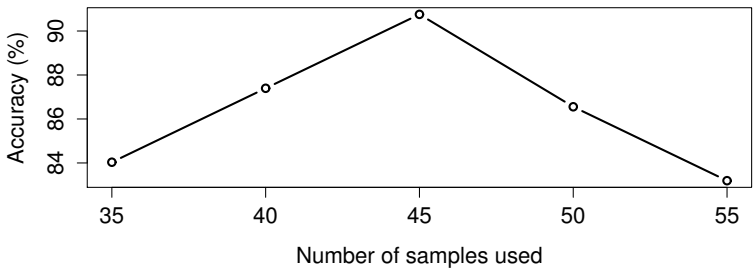
(a)



(b)

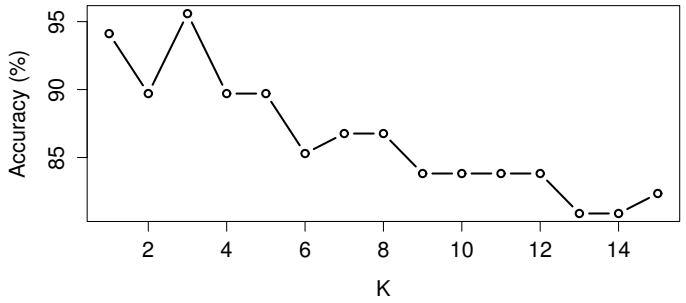


(c)

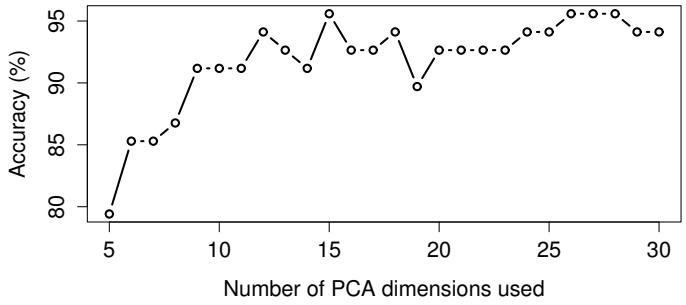


(d)

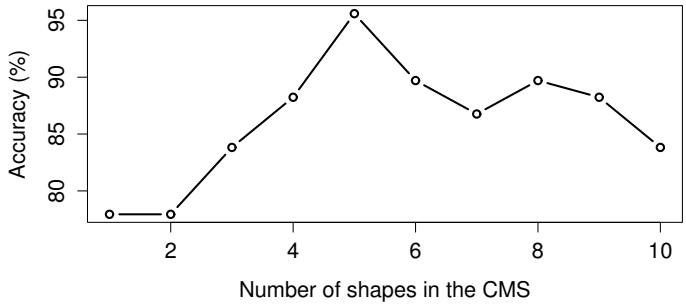
Figure 5.13: Accuracy curves of MuHAVi SVM tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.



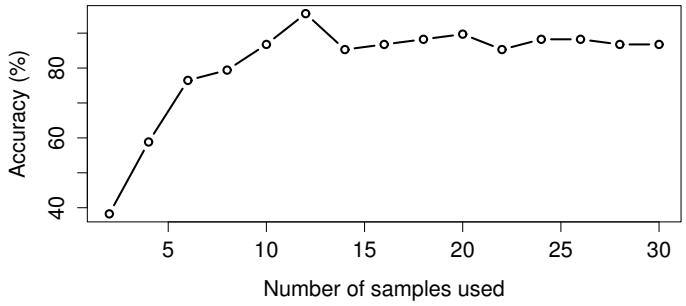
(a)



(b)



(c)



(d)

Figure 5.14: Accuracy curves of MuHAVi14 K-NN tests varying parameter (a) K; (b) PCA dimensions; (c) CMS number; (d) number of samples.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	.875	.125	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	.5	.5	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 5.23: Confusion matrix of the K-NN results for the MuHAVi14 data set.

Code	Action class	Code	Action class
1	Collapse left	8	Run right to left
2	Collapse right	9	Stand up left
3	Guard to kick	10	Stand up right
4	Guard to punch	11	Turn back left
5	Kick right	12	Turn back right
6	Punch right	13	Walk left to right
7	Run left to right	14	Walk right to left

Table 5.24: Action codes for MuHAVi14 confusion matrices.

SVM

The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.25. Based on these values, accuracies varying *cost*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.15(a) to (d), respectively.

Parameter	Value
Cost	100
PCA dimensions	18
CMS number	6
Number of samples	12

Table 5.25: Optimum parameters for MuHAVi data set using SVM classifier.

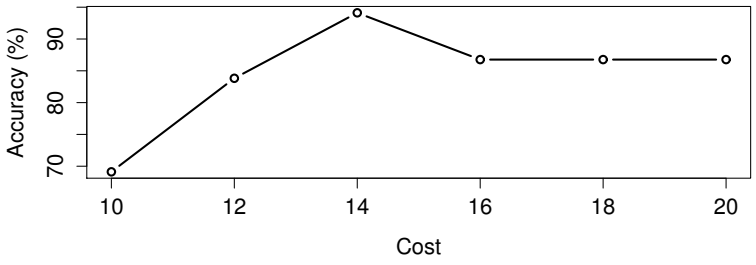
The confusion matrix is given in Table 5.26, with the action codes specified in Table 5.27.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	.75	0	.25	0	0	0	0	0	0	0	0	0	0
3	0	0	.875	.125	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0
9	.5	0	0	0	.5	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1

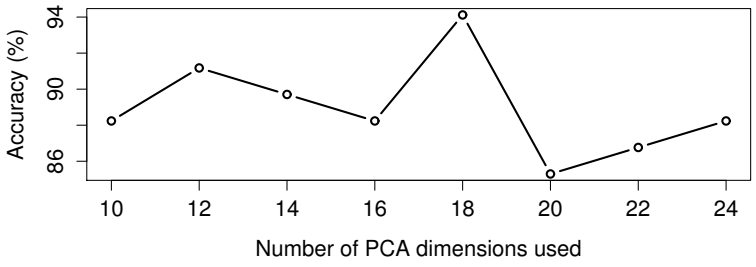
Table 5.26: Confusion matrix of the SVM results for the MuHAVi14 data set.

Code	Action class	Code	Action class
1	Collapse left	8	Run right to left
2	Collapse right	9	Stand up left
3	Guard to kick	10	Stand up right
4	Guard to punch	11	Turn back left
5	Kick right	12	Turn back right
6	Punch right	13	Walk left to right
7	Run left to right	14	Walk right to left

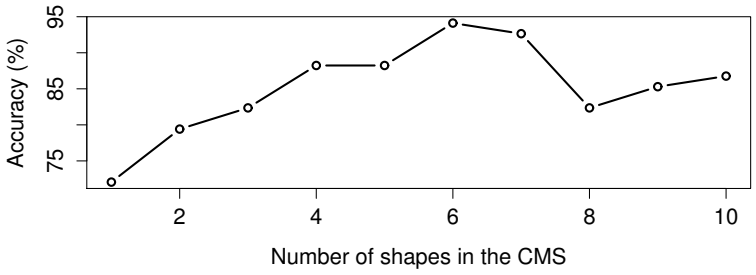
Table 5.27: Action codes for MuHAVi14 confusion matrices.



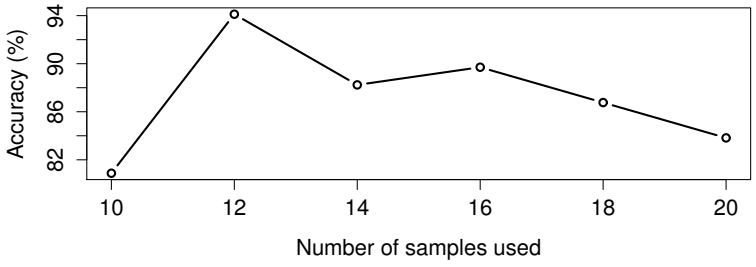
(a)



(b)



(c)



(d)

Figure 5.15: Accuracy curves of MuHAVi14 SVM tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.

5.3.3 MuHAVi8

The following results correspond to the MuHAVi8 data set for both K-NN and SVM classifiers.

K-NN

The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.28. Based on these values, accuracies varying *K*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.16(a) to (d), respectively.

Parameter	Value
K	1
PCA dimensions	12
CMS number	3
Number of samples	10

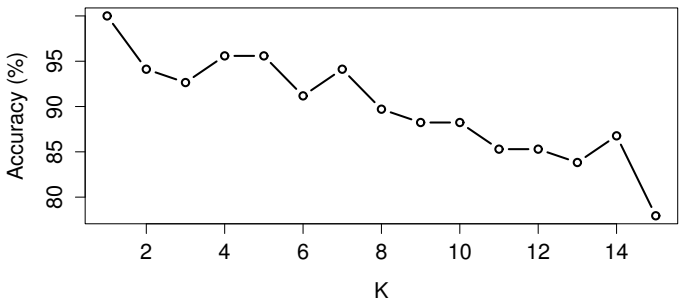
Table 5.28: Optimum parameters for MuHAVi8 data set using K-NN classifier.

SVM

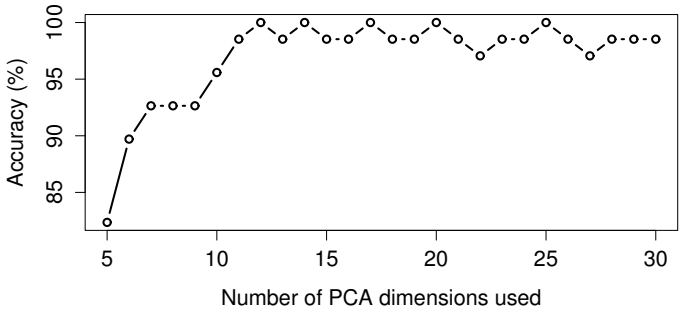
The parameters for the best accuracy, shown in Table 5.15, are presented in Table 5.29. Based on these values, accuracies varying *cost*, *PCA dimensions*, *CMS number*, and *number of samples* are shown in Figures 5.17(a) to (d), respectively.

Parameter	Value
Cost	100
PCA dimensions	22
CMS number	3
Number of samples	14

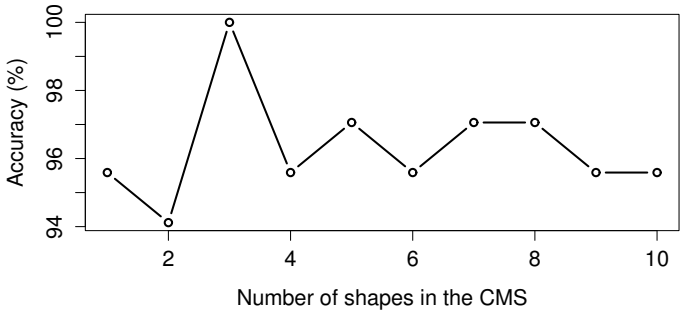
Table 5.29: Optimum parameters for MuHAVi data set using SVM classifier.



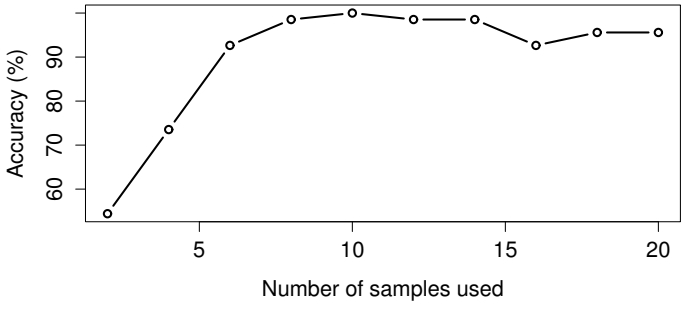
(a) K



(b) PCA dimensions

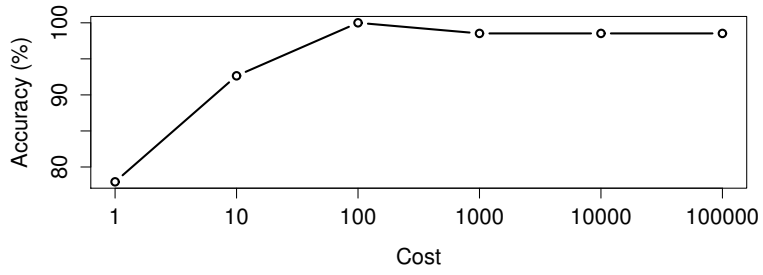


(c) CMS number

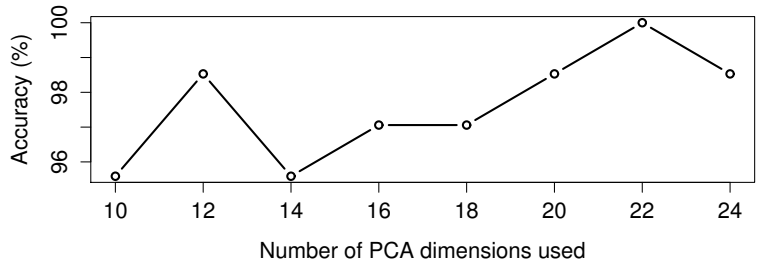


(d) Number of samples

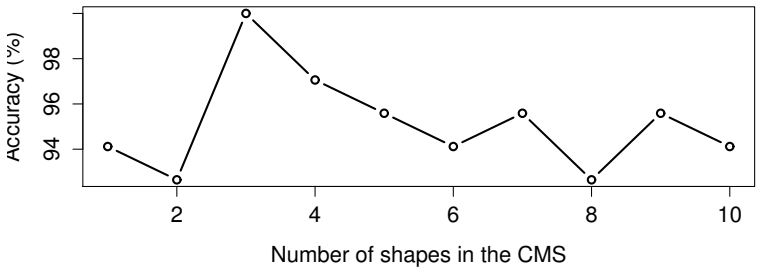
Figure 5.16: Accuracy curves of MuHAVi8 K-NN tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.



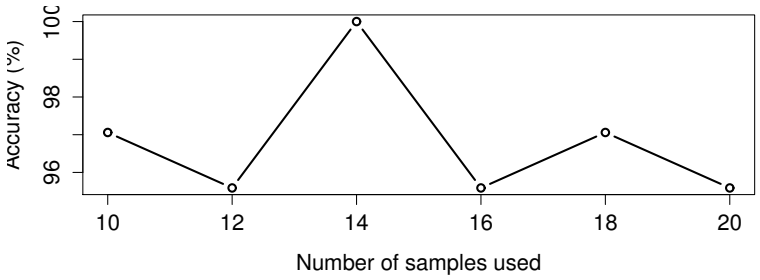
(a)



(b)



(c)



(d)

Figure 5.17: Accuracy curves of MuHAVi8 SVM tests varying parameter (a) cost; (b) PCA dimensions; (c) CMS number; (d) number of samples.

5.4 Discussion

This section briefly discusses the complexity analysis of the proposed method, the accuracy results for each tested data set and the granularity adjustment.

5.4.1 Complexity Analysis

The complexity of the feature extraction method depends on each part of the method, shown in Figure 4.1. In step (a), both the methods employed for background subtraction and frame difference are linear in relation to the number of pixels in the image. For every new frame, a new CMS is computed by uniting the whole temporal window, making the complexity of this step defined by the number of pixels multiplied by the CMS number. Step (b) includes sweeping the bounding box, so it is linear in relation to the size of the bounding box. Step (c) also includes sweeping the bounding box, so it is also linear with respect to the size of the bounding box. Step (d) is linear in relation to the number of interest points adopted. The classification stage is considered apart, because it is not part of this work contribution, such that any classifier could be used, making the overall time variable.

At first, obtaining the descriptor from a video frame has complexity $O(ns + b + i)$, where s is the size of the frame, n is the the CMS number, b is the bounding box size, and i is the number of interest points. However, i must be much smaller than b, (since it is on its borders), which must be smaller than s. Therefore, the overall complexity is $O(ns)$.

5.4.2 Accuracy Results

The method showed state-of-the-art accuracy for all tested data sets. On Weizmann, 97.85% was achieved. The descriptor was computed in an average of 1.175 FPS, which is less than one millisecond per frame. Most of the confusion is on the side and skip classes. On the manually segmented subset, a loss of accuracy was observed when using the SVM classifier. A possible reason is the effect explained in Section 4.1 and shown in Figure 4.2, where poor segmentation may emphasize the most important region of movement. This is confirmed from the confusion matrices, which show misclassification between waving one hand and two hands, which did not exist on the original set. The distance between the classes is shortened since the bounding boxes now comprise the whole body silhouette, resulting in less relevance to the position of the arms.

On KTH data set, the best result was 90.28% accuracy. The descriptor was computed at 215.02 FPS, which is less than five milliseconds per frame. The precision is a little inferior to the most recent state-of-the-art because of the difficulties in the movement

extraction in the set. From the confusion matrices, it can be seen that most of the confusion happens in two groups. One is the group of actions that consist of hand movement – *clapping*, *waving hands* and *boxing*. The other group, presenting more confusion, is the group of actions that comprise the displacement movements – *jogging*, *running* and *walking*. Most of the confusion happens between the actions *run* and *jog*. It is easy to understand that, since the two actions are very similar to each other.

Despite the noise movement present in the original MuHAVi data set, the accuracy achieved was 91.60%. This may be due to the multi training – instead of extracting one descriptor per video, as often done – where the noise is trained with the class label, but the classifier is often able to handle outliers. Since the set has the largest frames and highest CMS number, it had the smallest frame rate: 47.04 FPS, or 21.26 milliseconds per frame. The smallest frame rate achieved was a result of the combination of the largest frame with the highest CMS number. The confusion matrices showed significant confusion in the *punch* and *pick up and throw object* classes, both involving strong arm movement. Other confusion occurrences were sparse.

The accuracy obtained was 95.59% for MuHAVi14 and 100% for MuHAVi8. These results were expected since both data sets are very simple. The data sets consist of the same videos, such that their run time were the same: 203.12 FPS, less than 5 milliseconds per frame. In both SVM and K-NN, half of the confusion lies in the *stand up left* action. In Table 5.26, there is confusion on the *collapse left* action, which presents essentially the same movement, however, in inverse order. In Table 5.23, there is confusion on the *stand up right* action, which is the same action, however, with a different direction.

Among the parameters varied in the grids, the CMS number indicates the size of the temporal window that unites the silhouettes. The results, represented in the plot images, showed that the CMS number adds relevant information to the shapes, resulting in increase in accuracy. Most of the data sets had optimum values higher than one, indicating that using CMS was better than not using it.

5.4.3 Granularity Adjustment

Granularity refers to the *complexity* of an action. Actions that occurs in a short time span or being periodical are simple actions, having low granularity. Taking a step has low granularity; walking is the periodical repetition of this action, hence it also has low granularity. *Jumping over a fence* is a high granularity action, since it consists of more complex and aperiodic movements that are done in more than one possible way and requiring more time to take place.

Results have shown that the CMS number is able to adjust the granularity of the actions. The MuHAVi data set has the highest granularity of the actions. The classes are

more complex than just *walking* or *moving limbs*, since they are unions of these simple movements (*walking* is part of most of them).

The CMS number was increased until it was able to describe the actions entirely. It consisted of a granularity calibration. This ability indicates that the method is robust with respect to the actions that can be learned.

Chapter 6

Conclusions and Future Work

Human action recognition is an open research area. There are many problems demanding solutions and plenty of room for improvement in the already obtained results. Many issues are concerned with the quality of the recognition, such as robustness to occlusions, view angle variation, multi-view, and inter-camera displacement. Other trends are concerned with real situation applicability, such as processing time, recognition of multiple simultaneous agents, and functioning in continuous video streams.

Chapter 3 highlights the two most common strategies for obtaining information from videos: the first is based on appearance descriptors extracted from the neighborhood of selected interest points, such as developed by Laptev (2005) and Dollár et al. (2005); the second builds descriptors from silhouettes extracted by foreground segmentation methods, such as frame difference or background subtraction.

In this dissertation, human action recognition was investigated to search for its applications and solutions available. Two methods were proposed, approaching different stages of the process. The first joins motion shapes – forming the CMS (Cumulative Motion Shapes) – adding temporal information on static shapes and to correct possible defects on extracted foreground. The second is a descriptor for the shapes; it is lightweight and easily scalable to work with multiple action instances on a single video sequence, since it is applied to each movement instance separately.

The interest point selection strategy picks extreme points with the objective of finding body extremities and important articulations. Hands, elbows, feet, knees and head contain the most distinguishable pose information. The extreme point strategy succeeds in finding these regions without making use of expensive matching algorithms.

Actions can be performed by different actors and with distinct starts. Because of that, correlating an order to CMS can be a complex task. Multi-training avoids such problem by using each CMS independently in the training process and considering each one such as a unique vote in the prediction judgment.

The method was tested on three commonly used data sets and produced state-of-the-art accuracy (Tables 3.1 and 3.2), even on KTH data set (Table 5.10), which contains unrealistic camera movement. Experiments also demonstrated proper results on MuHAVi (Table 5.15), which is a difficult data set, since some sequences contain people walking around in the background or arranging the scenario and most videos include parts when the actor performs actions different from the one labeled (mostly walking), increasing inter-class similarities along the sequences. Finally, the method obtained high accuracy rates for Weizmann data set (Table 5.1) with impressive speed rate.

The plots that present the change of accuracy varying the CMS number showed that, in most of the cases, the optimum value is higher than one – value one means that the original shapes are used – even for manually annotated silhouettes, with accuracy difference of up to 20% (for MuHAVi14, Figure 5.15). This means that CMS adds positive information, which results in enhancement in the classification using the proposed descriptor.

The best CMS numbers for MuHAVi data set are considerably higher than for other data sets. This is because the granularity of its actions is very high. The classes are more complex than just walking or moving limbs, they are unions of these simple movements; walking is a part of most of them. Accumulating higher numbers of shapes allowed the calibration of the granularity of the actions trained, indicating that the proposed method is robust in relation to the actions that can be learned.

As a shape-based descriptor, it naturally requires the actions to be distinguishable through shapes. Hence, it might be unsuitable for classification of, for example, mouse behavior, as it appears on the data set of Dollár et al. (2005), or kissing, as in the Hollywood data set (Laptev et al. 2008).

Some directions for future work include:

- *the evaluation of the method in other data sets:* the method can be validated on actions of different nature. Testing the method in other sets allows us to have better understanding of the effectiveness of the work. It is desirable to obtain results in other data sets, such as Microsoft Research (MSR) action data set (Liu 2012), UCF sports action data set (Rodriguez et al. 2008), and UCF50 Action Recognition data set (Reddy & Shah 2013).

- *extending the method to more complex scenarios:* the process stages targeted in this dissertation are the backbone of an action recognition system, however, there are more challenges to the matter. One is the application of a system in continuous videos. New problems appear, such as identification of when an actor stops doing an action and starts doing another one. A desirable capability is the detection of multiple simultaneous actors, which involves using effective tracking methods.

- *multicamera:* it consists of using multiple cameras to identify actions. In a given time, an actor may or may not be visible from more than one camera, so the system must be able to identify these occurrences. Tracking and identification of people are tools for the problem. This may even enable the system to be aware of what trajectory a person is following.

Bibliography

Alcântara, M. F., Moreira, T. P. & Pedrini, H. (2013). Motion Silhouette-Based Real Time Action Recognition, *18th Iberoamerican Congress on Pattern Recognition*, Vol. 8259, Lecture Notes in Computer Science, Havana, Cuba, pp. 471–478.

Alcântara, M. F., Moreira, T. P. & Pedrini, H. (2014). Real-time action recognition based on cumulative motion shapes, *Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, pp. 2941–2945.

Barron, J. L., Fleet, D. J. & Beauchemin, S. S. (1994). Performance of Optical Flow Techniques, *International Journal of Computer Vision* **12**(1): 43–77.

Baum, L. E. & Petrie, T. (1966). Statistical Inference for Probabilistic Functions of Finite State Markov Chains, *Annals of Mathematical Statistics* **37**: 1554–1563.

Blank, M., Gorelick, L., Shechtman, E., Irani, M. & Basri, R. (2005). Actions as Space-Time Shapes, *International Conference on Computer Vision*, Beijing, China, pp. 1395–1402.

Bregonzio, M., Xiang, T. & Gong, S. (2012). Fusing Appearance and Distribution Information of Interest Points for Action Recognition, *Pattern Recognition* **45**(3): 1220–1234.

Chaaraoui, A., Climent-Pérez, P. & Flórez-Revuelta, F. (2013). Silhouette-based Human Action Recognition using Sequences of Key Poses, *Pattern Recognition Letters* **34**(15): 1799 – 1807. Smart Approaches for Human Action Recognition.

Chaaraoui, A. & Flórez-Revuelta, F. (2013). Human Action Recognition Optimization based on Evolutionary Feature Subset Selection, *Genetic and Evolutionary Computation Conference*, New York, NY, USA, pp. 1229–1236.

Cheema, S., Eweiwi, A., Thurau, C. & Bauckhage, C. (2011). Action Recognition by Learning Discriminative Key Poses, *International Conference on Computer Vision*, Barcelona, Spain, pp. 1302–1309.

Chen, M. & Hauptmann, A. (2009). MoSIFT: Recognizing Human Actions in Surveillance Videos, *Technical report*, Carnegie Mellon University Computer Science, Forbes Avenue Pittsburgh, PA, USA.

Cortes, C. & Vapnik, V. (1995). Support-Vector Networks, *Machine Learning* **20**(3): 273–297.

Cover, T. & Hart, P. (1967). Nearest Neighbor Pattern Classification, *IEEE Transactions on Information Theory* **13**(1): 21–27.

Deans, S. (2007). *The Radon Transform and Some of Its Applications*, Dover Books on Mathematics Series, Dover Publications.

Dollár, P., Rabaud, V., Cottrell, G. & Belongie, S. (2005). Behavior Recognition via Sparse Spatio-Temporal Features, *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65–72.

Guo, K., Ishwar, P. & Konrad, J. (2013). Action Recognition From Video Using Feature Covariance Matrices, *Image Processing* **22**(6): 2479–2494.

Haritaoglu, I., Harwood, D. & Davis, L. (2000). W4: Real-Time Surveillance of People and their Activities, *Pattern Analysis and Machine Intelligence* **22**(8): 809–830.

Hartigan, J. A. & Wong, M. A. (1979). Algorithm AS 136: A K-Means Clustering Algorithm, *Applied Statistics* **28**(1): 100–108.

Herbert, B., Andreas, E., Tuytelaars, T. & Van Gool, L. (2008). Speeded-Up Robust Features (SURF), *Computer Vision and Image Understanding* **110**(3): 346–359.

Hsieh, C. H., Huang, P. & Tang, M. D. (2011). The Recognition of Human Action Using Silhouette Histogram, *in* M. Reynolds (ed.), *Australasian Computer Science Conference*, Vol. 113, Perth, Australia, pp. 11–16.

Hu, W., Tan, T., Wang, L. & Maybank, S. (2004). A Survey on Visual Surveillance of Object Motion and Behaviors, *Systems, Man, and Cybernetics, Part C: Applications and Reviews* **34**(3): 334–352.

Jacques Junior, J., Musse, S. & Jung, C. (2010). Crowd Analysis Using Computer Vision Techniques, *IEEE Signal Processing Magazine* **27**(5): 66–77.

Ji, S., Xu, W., Yang, M. & Yu, K. (2013). 3D Convolutional Neural Networks for Human Action Recognition, *Pattern Analysis and Machine Intelligence* **35**(1): 221–231.

Jolliffe, I. (2002). *Principal Component Analysis*, 2nd edn, Springer.

Junejo, I. N. & Aghbari, Z. A. (2012). Using SAX Representation for Human Action Recognition, *Journal of Visual Communication and Image Representation* **23**(6): 853–861.

Kaewtrakulpong, P. & Bowden, R. (2001). An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection, *European Workshop on Advanced Video Based Surveillance Systems*, Vol. 5308, London, UK.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems, *Transactions of the ASME - Journal of Basic Engineering* **82**(Series D): 35–45.

Karthikeyan, S., Gaur, U., Manjunath, B. S. & Grafton, S. (2011). Probabilistic Subspace-based Learning of Shape Dynamics Modes for Multi-view Action Recognition, *International Conference on Computer Vision*, Barcelona, Spain, pp. 1282–1286.

Khotanzad, A. & Hong, Y. (1990). Invariant Image Recognition by Zernike Moments, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(5): 489–497.

Laptev, I. (2005). On Space-Time Interest Points, *International Journal of Computer Vision* **64**(2-3): 107–123.

Laptev, I., Marszalek, M., Schmid, C. & Rozenfeld, B. (2008). Learning Realistic Human Actions from Movies, *Computer Vision and Pattern Recognition*, pp. 1–8.

Lin, T. & Zhang, H.-J. (2000). Automatic Video Scene Extraction by Shot Grouping, *International Conference on Pattern Recognition*, Vol. 4, pp. 39–42 vol.4.

Liu, Z. (2012). Microsoft Research Action Data Sets. `http://research.microsoft.com/en-us/um/people/zliu/ActionRecoRsrc/`.

Lowe, D. (1999). Object Recognition from Local Scale-Invariant Features, *Computer Vision*, Vol. 2, pp. 1150–1157 vol.2.

McKenna, S. J., Jabri, S., Duric, Z., Rosenfeld, A. & Wechsler, H. (2000). Tracking Groups of People, *Computer Vision and Image Understanding* **80**(1): 42–56.

Moghaddam, B. (2002). Principal Manifolds and Probabilistic Subspaces for Visual Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(6): 780–788.

Moghaddam, Z. & Piccardi, M. (2010). Histogram-Based Training Initialisation of Hidden Markov Models for Human Action Recognition, *International Conference on Advanced Video and Signal Based Surveillance*, Boston, MA, USA, pp. 256–261.

Moghaddam, Z. & Piccardi, M. (2013). Training Initialization of Hidden Markov Models in Human Action Recognition, *Automation Science and Engineering* **36**(99): 1–15.

Onofri, L. & Soda, P. (2012). Combining Video Subsequences for Human Action Recognition, *International Conference on Pattern Recognition*, Tsukuba, Japan, pp. 597–600.

Pearson, K. (1901). Principal Component Analysis, *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **6**(2): 559.

Pedrini, H. & Schwartz, W. (2007). *Análise de Imagens Digitais: Princípios, Algoritmos e Aplicações*, Thomson-Learning, chapter 9.

Raja, K., Laptev, I., Perez, P. & Oisel, L. (2011). Joint Pose Estimation and Action Recognition in Image Graphs, *International Conference on Image Processing*, Brussels, Belgium, pp. 25–28.

Reddy, K. K. & Shah, M. (2013). Recognizing 50 Human Action Categories of Web Videos, *Machine Vision Applications* **24**(5): 971–981.

Rodriguez, M., Ahmed, J. & Shah, M. (2008). Action MACH a Spatio-Temporal Maximum Average Correlation Height Filter for Action Recognition, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8.

Ryoo, M. S. & Aggarwal, J. K. (2009). Spatio-Temporal Relationship Match: Video Structure Comparison for Recognition of Complex Human Activities, *International Conference on Computer Vision*, Kyoto, Japan, pp. 1593–1600.

Sasse, M. A. (2010). Not Seeing the Crime for the Cameras?, *Communications of the ACM* **53**(2): 22–25.

Schuldt, C., Laptev, I. & Caputo, B. (2004). Recognizing Human Actions: A Local SVM Approach, *17th International Conference on Pattern Recognition*, Vol. 3, Cambridge, UK, pp. 32–36 Vol.3.

Singh, S., Velastin, S. A. & Ragheb, H. (2010). MuHAVi: A Multicamera Human Action Video Dataset for the Evaluation of Action Recognition Methods, *Advanced Video and Signal Based Surveillance*, pp. 48–55.

Stauffer, C. & Grimson, W. (1999). Adaptive Background Mixture Models for Real-Time Tracking, *Computer Vision and Pattern Recognition*, Vol. 2, Ft. Collins, CO, USA, pp. 2 vol. (xxiii+637+663).

Sun, X., Chen, M. & Hauptmann, A. (2009). Action Recognition via Local Descriptors and Holistic Features, *Computer Vision and Pattern Recognition*, Miami, FL, USA, pp. 58–65.

Ta, A.-P., Wolf, C., Lavoue, G., Baskurt, A. & Jolion, J. M. (2010). Pairwise Features for Human Action Recognition, *International Conference on Pattern Recognition*, Istanbul, Turkey, pp. 3224–3227.

Tishby, N., Pereira, F. & Bialek, W. (1999). The Information Bottleneck Method, *Allerton Conference on Communication, Control and Computing*, pp. 368–377.

Turaga, P., Chellappa, R., Subrahmanian, V. S. & Udrea, O. (2008). Machine Recognition of Human Activities: A Survey, *Circuits and Systems for Video Technology* **18**(11): 1473–1488.

Wang, S., Huang, K. & Tan, T. (2009). A Compact Optical Flowbased Motion Representation for Real-Time Action Recognition in Surveillance Scenes, *International Conference on Image Processing*, Cairo, Egypt, pp. 1121–1124.

Wiskott, L. & Sejnowski, T. (2002). Slow Feature Analysis: Unsupervised Learning of Invariances, *Neural Computation* **14**(4): 715–770.

Wu, C., Khalili, A. H. & Aghajan, H. (2010). Multiview Activity Recognition in Smart Homes with Spatio-Temporal Features, *International Conference on Distributed Smart Cameras*, Atlanta, Georgia, pp. 142–149.

Yang, M., Lv, F., Xu, W., Yu, K. & Gong, Y. (2009). Human Action Detection by Boosting Efficient Motion Features, *IEEE 12th Conference on Computer Vision Workshops*, pp. 522–529.

Zhang, Z. & Tao, D. (2012). Slow Feature Analysis for Human Action Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**(3): 436–450.