Real-Time Human Action Recognition Using a Reduced Feature Set

by

Gloria Castro Muñoz

Dissertation
Submitted to the Electronics Science Department
In Partial Fulfillment of the Requirements for the Degree of

PH.D. IN ELECTRONICS SCIENCE

at the

National Institute for Astrophysics, Optics and Electronics
December 2015
Tonantzintla, Puebla, México

Supervised by

Ph.D. Jorge Francisco Martínez Carballido

Dedication

To my parents.

Thanks for teach me the ethics and discipline that today guide my life.

To my husband.

Thanks for your unconditional love and support.

Acknowledgements

I would like to express my gratitude to my supervisor Professor *Jorge Martínez Carballido* for his unwavering support, time, and mentorship through this project.

Thanks to CONACYT for the financial support granted through the scholarship for doctoral studies number 328836.

Abstract

The Human Action Recognition (HAR) from video sequences is a topic which has captured the interest of a large number of researchers from industry, academia, consumer agencies and security agencies. The solid interest in the topic is motivated by the wide variety and importance of promising applications for example rehabilitation of patients, monitoring and supporting of children and elderly people, automatic annotation of video, human-computer interfaces, and video surveillance among others.

Particularly the increasing demand for security and safety by society in recent years has occasioned significant advances in video surveillance technology. However despite these advances, video surveillance systems are not able to analyze in real-time the huge amounts of video coming from video surveillance cameras installed in the worldwide and therefore, they can't detect and alert about potential criminal activity in real-time.

Faced with this situation, it is anticipated that video surveillance systems will migrate to autonomous video analysis "on the edge"; where algorithms, embedded on a surveillance camera, will analyze retrieved video in real-time and autonomously from its field of vision in search of unwanted events; so that the system can advise to an upper instance to take action.

A first step towards this autonomous analysis is the recognition of basic human actions. Therefore this dissertation presents a *real-time HAR method* based on simple techniques that incorporate information of natural domain knowledge of the problem to achieve an efficient recognition with attributes such as *simplicity, precision and speed*. The method has four main stages: bounding box tracking,

bounding box representation, features extraction and action classification. The method was implemented on two development platforms (*Matlab and C++*) and evaluated on three publicly available datasets: *Weizmann*, *UIUC* and *i3DPost*. Results obtained on the three datasets show that the proposed method is superior to other state of the art methods, in *processing capacity* (18,282 fps, 2,427 fps and 742 fps, respectively) and comparable or superior in *accuracy* (99.95%, 100%, and 99%). The method is the first one that shows *real-time* performance on video up *8K UHD*. In addition, since the presented method is based on *clear* and *simple concepts* and shows recognition skills on *real-time*, it is foreseen that it could be embedded on a camera allowing the so-called *"edge processing"*.

Resumen

El reconocimiento de acciones humanas (HAR) a partir de secuencias de video, es un tema que ha atraído el interés de un gran número de investigadores en la industria, academia, agencias del consumidor y agencias de seguridad. El fuerte interés en el tema está sustentado en la importancia y amplia variedad de las aplicaciones potenciales HAR, por ejemplo rehabilitación de pacientes, anotación automática de video, monitoreo y apoyo a niños y personas mayores, interfaces humano-computadora y video vigilancia entre otras.

Particularmente la demanda creciente en años recientes de seguridad y protección por parte de la sociedad ha ocasionado avances significativos en la tecnología de video vigilancia. Sin embargo, a pesar de esos avances, los sistemas de video vigilancia aún no son capaces de analizar en tiempo real la enorme cantidad de video proveniente de cámaras de video vigilancia instaladas en todo el mundo y por lo tanto, no pueden detectar y emitir alertas en tiempo real en el caso de existir actividad criminal potencial.

Ante esta situación, se prevé que los sistemas de video vigilancia convencionales migrarán al análisis del video autónomo "en el borde"; donde algoritmos embebidos en una cámara de video vigilancia, analizarán de forma autónoma y en tiempo real el video obtenido de su campo visual en busca de eventos no deseados; tal que el sistema pueda advertir a una instancia superior y de esta forma tomar acciones.

Un primer avance hacia ese análisis autónomo es el reconocimiento de acciones humanas básicas. Por lo tanto esta tesis presenta un método HAR en tiempo real basado en técnicas simples que incorporan información del conocimiento en el dominio natural del problema para obtener un reconocimiento eficiente con atributos

tales como simplicidad, precisión y velocidad. El método consta de cuatro etapas principales: Seguimiento del objeto de interés, representación de la figura humana, extracción de rasgos y clasificación de acción.

El método fue implementado en dos plataformas de desarrollo (Matlab y C++) y evaluado sobre tres conjuntos de acciones disponibles públicamente: Weizmann, UIUC e i3DPost. Los resultados obtenidos con estos tres conjuntos de acciones muestran que el método propuesto es superior a otros métodos del estado del arte en capacidad de procesamiento (18,282 fps, 2,427 fps y 742 fps respectivamente) y comparable o superior en precisión (99.95%, 100% y 99%). El método presentado es el primero que muestra rendimiento en tiempo real en videos de hasta 8K UHD. Adicionalmente; debido a que el método está basado en conceptos claros, simples y muestra atributos en tiempo real, se prevé que este puede ser embebido en una cámara y permitir el procesamiento del video “en el borde”.

Publications

- **Gloria Castro-Muñoz**, Jorge Martínez-Carballido, Roberto Rosas-Romero, "A novel reduced feature set and a hierarchical system of classifiers for human action recognition based on the natural domain knowledge of the human figure". In *Power, Electronics and Computing (ROPEC), 2014 IEEE International Autumn Meeting on*, pp.1-6, 5-7 Nov. 2014. Doi: 10.1109/ROPEC.2014.7036338. (**Best Paper Award**)

- **Gloria Castro-Muñoz**, Jorge Martínez-Carballido, Roberto Rosas-Romero, "A human action recognition approach with a novel reduced feature set based on the natural domain knowledge of the human figure". *Signal Processing: Image Communication*, Volume 30, January 2015, Pages 190-205, ISSN 0923-5965. Doi:10.1016/j.image.2014.10.002.

- **Gloria Castro-Muñoz** and Jorge Martínez-Carballido, "Real-Time Human Action Recognition Using Full and Ultra High Definition Video". In *Computational Science and Computational Intelligence (CSCI), 2015 International Conference on*, December 7-9, 2015. Las Vegas USA. Manuscript submitted and accepted for publication.

Contents

List of Abbreviations ..... **ix**

List of Figures ..... **xi**

List of Tables ..... **xiii**

List of Algorithms ..... **xiv**

Glossary ..... **xv**

1 Introduction **1**

1.1 Background 1
1.2 Problem Statement 3
1.3 Dissertation Goals 4
1.4 Dissertation Contribution and Organization 5

2 Related Work **6**

2.1 HAR Methods 6
2.1.1 Discussion 8
2.2 Real-Time HAR Methods 9
2.3 Resolution Video in HAR methods 11
2.4 Summary 14

3 Theory Fundamentals **15**

3.1 Human Action Recognition (HAR) 15
3.1.1 Motion Hierarchy 15
3.1.2 HAR Processing Stages 16
3.2 HAR Evaluation 17

3.2.1 Cross-Validation (CV) 17
3.2.2 Confusion Matrix 19
3.3 Object Segmentation 20
3.3.1 Background Subtraction 20
3.3.2 Otsu Thresholding Method 21
3.4 Classification 22
3.4.1 Perceptron 22
3.4.2 Support Vector Machine (SVM) 24
3.5 Summary 28

4 Proposed HAR Method 29

4.1 Overview of the Method 29
4.2 Method Evolution 31
4.3 Pre-processing 32
4.3.1 Silhouette Extraction 32
4.3.2 Generation of Snippet Sequences 34
4.4 Bounding Box Tracking 35
4.5 Bounding Box Representation 37
4.6 Features Extraction 38
4.6.1 Local Features 41
4.6.2 Global Features 42
4.7 Classifier 44
4.7.1 Architecture 45
4.7.2 Training 46
4.7.3 Testing 46
4.8 Summary 48

5 Experiments and Results 49

5.1 Datasets 49
5.1.1 Weizmann Dataset 50
5.1.2 University of Illinois at Urbana-Champaign (UIUC) Dataset 50
5.1.3 i3DPost Multi-View Dataset 51
5.2 Experimental Setup 52
5.3 HAR Method Based in Four Features 53
5.3.1 Accuracy Evaluation 53
5.3.2 Speed Evaluation 54

5.3.3 Comparison with other Methods 55
5.4 HAR Method Based in Six Features 56
5.4.1 Accuracy Evaluation 56
5.4.2 Speed Evaluation 58
5.4.3 Comparison with other Methods 59
5.5 Importance of Pre-processing Stage 60
5.6 Importance of Bounding Box Tracking 64
5.7 Multi-Dataset Evaluation on C++ 66
5.7.1 Accuracy 67
5.7.2 Speed 68
5.7.3 Comparison 71
5.8 Complexity Analysis 74
5.9 Multi-Resolution Timing Evaluation 76
5.10 Summary 78

6 Conclusions and Future Work 80

6.1 Conclusions 80
6.2 Future Work 81

A. Video Surveillance Systems in Mexico City 83

B. Market Growth of HD Surveillance Cameras 85

C. Data Growth of Video Surveillance 86

Bibliography 87

List of Abbreviations

AAL	Ambient Assisted Living
BB	Bounding Box
BOW	Bags of Words
BS	Background Subtraction
BSVM	Bias Support Vector Machine
CCR	Correct Classification Rate
CCTV	Closed Circuit Television
CV	Cross Validation
DTW	Dynamic Time Warping
FB	Feet Box
FPS	Frames Per Second
GMM	Gaussian Mixture Models
HAR	Human Action Recognition
HD	High Definition
HOR	Histogram of Oriented Rectangles
HVT-HMM	Hierarchical Variable Transition Hidden Markov Model
KB	Knee Box
LOAO	Leave One Actor Out

LOOCV	Leave One Out Cross Validation
LOSO	Leave One Sequence Out
LPO	Leave P Out
MEI	Motion Energy Image
MHI	Motion History Image
MLD	Moving Light Display
OpenCV	Open Source Computer Vision
PaHOF	Pyramid of Accumulated Histograms of Optical Flow
PHOG	Pyramid of Histogram of Oriented Gradient
pLSA	Probabilistic Latent Semantic Analysis
ROI	Region Of Interest
SVM	Support Vector Machine
TDIRBF	Time Delay Input Radial Basis Function Network
UHD	Ultra High Definition
UIUC	University of Illinois at Urbana-Champaign
VFCV	V-Fold Cross Validation

List of Figures

Figure 3.1. Diagram of motion hierarchy. 16
Figure 3.2. Main processing stages of HAR method. 16
Figure 3.3. Hierarchy of some common types of Cross-Validation. 17
Figure 3.4. Model of a basic perceptron. 22
Figure 3.5. Region of decision for the case of two linearly separable classes. 23
Figure 3.6. Optimal hyperplane for two linearly separable classes. 24
Figure 3.7. Example of non-separable patterns. 25
Figure 4.1. The workflow of the proposed HAR method. 30
Figure 4.2. Sub-processes characteristic of the HAR method based on four features. 31
Figure 4.3. Examples of silhouettes extracted from the Weizmann dataset. 33
Figure 4.4. Examples of silhouettes extracted from UIUC dataset. 33
Figure 4.5. Examples of silhouettes obtained for i3DPost dataset. 34
Figure 4.6. Outline of the BB tracker. 36
Figure 4.7. Model of human body based in three boxes: BB, KB and FB. 38
Figure 4.8. Standard geometrical proportions of human body. 39
Figure 4.9. Examples of domain knowledge for person Lena of Weizmann dataset. 40
Figure 4.10. Hierarchical system of classifiers for the proposed method. 45
Figure 4.11. Top-down hierarchy of classification. 46
Figure 4.12. Procedure of action-labeling. 47
Figure 5.1. Examples of frames extracted from Weizmann dataset. 50
Figure 5.2. Examples of frames extracted from UIUC dataset. 51
Figure 5.3. Examples of frames extracted from i3DPost dataset. 51
Figure 5.4. Confusion matrix obtained for the Weizmann dataset using the LOAO Protocol (left) and Protocol 60% — 40% (right). 53
Figure 5.5. Confusion matrix obtained for the UIUC dataset using the LOAO Protocol (left) and Protocol 60% — 40% (right). 54

Figure 5.6. Confusion matrix obtained for the Weizmann dataset using the LOAO Protocol (left) and Protocol 60%-40% (right)..... 57
Figure 5.7. Confusion matrix obtained for the UIUC dataset using the LOAO Protocol (left) and Protocol 60%-40% (right)..... 57
Figure 5.8. Percentage contributions of each stage to the average run time per snippet of the proposed HAR method for Weizmann and UIUC datasets..... 59
Figure 5.9. Examples of abnormal silhouettes extracted for three datasets, Weizmann (Rows 1-2), UIUC (rows 3-4) and i3DPost (rows 5-6). 61
Figure 5.10. Percentages of normal and abnormal silhouettes. 62
Figure 5.11. Confusion matrix obtained for Protocol 40% - 60% with normal silhouettes (40% of all snippets) used for training and abnormal silhouettes (60% of all snippets) used for testing. 63
Figure 5.12. Decision maps with critical snippet feature vectors corresponding to misclassified actions. 63
Figure 5.13. Examples of errors in BB tracking introduced by incorrect ω values such as "hands out" (left part and middle part) and "feet out" in (right part). 64
Figure 5.14. Confusion matrix using the LOOCV protocol with errors introduced by the BB tracking. 65
Figure 5.15. Confusion matrix obtained for a) Weizmann, b) UIUC and c) i3DPost datasets using LOAO..... 67
Figure 5.16. Percentage contributions by stage to the average run time per snippet. 69
Figure 5.17. Percentage contributions by stage to average run time per frame. 70
Figure 5.18. Bounding Box search area. 75
Figure 5.19. Bounding Box frame, size, and time relations..... 76
Figure 5.20. Resizing of i3DPost dataset. 77
Figure 5.21. Percentage contributions of each stage to average run time for different i3DPost dataset resolutions..... 78
Figure A.1. C4i4 Mexico City 84
Figure B.1. Expected worldwide growing of surveillance camera sales 85
Figure C.1. Global data generated daily by surveillance cameras..... 86

List of Tables

Table 2.1. Summary of HAR methods showing real-time performance. 12
Table 2.2. Review of real-time HAR methods and the video resolution used. 13
Table 3.1. Confusion Matrix for standard two-class problem. 19
Table 3.2. Evaluation measures obtained from confusion matrix. 19
Table 4.1. Values of the parameters involved in computation of ω 37
Table 5.1. Results of average run time evaluation per snippet. 54
Table 5.2. Comparison of the proposed method with other methods for Weizmann dataset. 56
Table 5.3. Comparison of the proposed method with other methods for UIUC dataset. 56
Table 5.4. Results of time evaluation per snippet for Weizmann and UIUC datasets. 58
Table 5.5. Comparison of the proposed method with other methods for Weizmann dataset. 60
Table 5.6. Percentage of occurrence for each type of abnormal silhouettes. 62
Table 5.7. Results of analysis for different ω values at the BB tracking for Weizmann, UIUC and i3DPost datasets. 66
Table 5.8. Results of average run time per snippet for three datasets on Matlab and C++. 68
Table 5.9. Meaningful results of proposed HAR method. 70
Table 5.10. Comparison of the proposed method with other methods for the Weizmann dataset. 72
Table 5.11. Comparison of the proposed method with other methods for the UIUC. 73
Table 5.12. Comparison of the proposed method with other methods for the i3DPost 73
Table 5.13. Time performance for different i3dpost dataset resolutions. 77

List of Algorithms

Algorithm 1: Silhouette extraction 33
Algorithm 2: Bounding Box Tracking 36
Algorithm 3 : Local Features Extraction 42
Algorithm 4 : Global Feature Extraction 44

Glossary

Chronophotography— Photograph or a series of photographs of a moving object taken to capture successive phases of the object's motion.

Classification— A general term for the assignment of a label (or class) to an input.

Domain knowledge— Knowledge relevant to a specific field of interest (environment, situation or problem).

Edge processing— Analytics and knowledge generation occur at the place where the data are collected, so that only the significant information is transported and stored.

Embedded system— A computer system with a specific function, often with real-time computing constraints.

Feature— A distinctive attribute derived of something or someone which can be represented as a numerical property.

Frame— Each of the pictures that make up a video.

Hyperplane— A geometrical construct which extends the idea of a plane in three dimensions to a general *d*-dimensional space.

Object tracking— The process of estimating the location in time of a moving object.

Perceptron—A computational element often used for classifying data into one of two classes.

Recognition— The process of associating some observations with a particular instance or class of object that is already known.

ROI— A subregion of an image where processing is to occur.

Snippet— A short sub-sequence of an entire video.

SVM— A classifier using supervised learning that is characterized by maximizing the distance between classes.

Chapter 1

1 Introduction

1.1 Background

The interest to understand human movement goes back to more than 2,000 years ago, when the motion was represented by means of static artwork; by such artists like Aristotle, da Vinci and Michelangelo characterized this first stage of development [**1**].

The art was the major driving force to understand human motion for many centuries until moving pictures appeared. It happened nearly 2000 years later, at the end of the 19th century, when chronophotography provided a new tool for understanding movement. Experiments of Janssen [**2**] and Muybridge [**3**] highlighted in this second stage of development.

In 1874, the French astronomer Pierre Janssen used a multi-exposure camera which took forty-eight exposures in seventy-two seconds for recording the transit of Venus across the Sun. This experiment was known as Janssen's revolver.

In 1878, the Bristish-born Eadweard Muybridge inspired by a dispute claim that a galloping horse may have all four hooves off ground set up a series of 12 cameras for recording fast motion alongside a barn. His experiment showed all four hooves off the ground for part of the time. He also invented a machine for displaying the

recorded series of images and applied it to human movement studies. Then, his experiments were very influential for the beginning of cinematography.

In the third stage, the motion analysis took a different perspective. The experiments instead of only capturing images of motion sequences, now wanted to capture data at higher level of abstraction which could then serve to reconstruct the original motion. For example, Marey [**4**] in 1894 abstracted the images of a runner to a system of bright lines for representing the positions of his limbs and he obtained sinuous curves of human gait. In 1891, Braune and Fischer [**5**] attached light rods to an actor's limbs over a black suit to quantitatively measure the human gait, and Johansson [**6**] in 1973 performed experiments using light indicator (Moving Light Display, MLD) placed on the body of human actors to study some human actions.

Finally, the advent of the computer and of digital technology in general in the latter half of the 20th century provided the tools for analyzing human motion based on digitized image sequences, and for animating or studying human motion using extensive calculations and detailed model of human locomotion.

Actually, Human Action Recognition (HAR) is a task that can be analyzed in two different ways, using external sensors and wearable sensors. In the first, the sensors are fixed in predetermined points of interest and in the latter; the sensors are attached to the subject. This work addressed HAR based in video sequences which is considered part of the analysis using external sensors. For greater detail about HAR using wearable sensors refer to [**7**].

HAR based in video sequences is a topic which has captured the interest of a large number of researchers from industry, academia, consumer agencies and security agencies. The great interest in HAR is supported by the wide variety of promising applications among which are: rehabilitation of patients [**8**], analysis and optimization of athletes performance [**9**], monitoring and supporting of patients, children and elderly people [**10**], Ambient-Assisted Living (AAL) [**11**], Automatic Annotation of Video [**12**], character animation in cinematographic recordings, Human-Computer Interfaces [**13**], and Video Surveillance [**14**] [**15**] among others. Particularly in security and safety areas Video Surveillance based in HAR is regarded as an important support tool.

1.2 Problem Statement

The increasing demand for security and safety by society have led the video surveillance technology to become essential around the world. One of the priorities of national security is the detection and prevention of criminal activity and terrorism right at the instant this happens. To achieve this goal, governments have opted to install conventional video surveillance systems in places of interest such as streets, government agencies, airports and train stations, among others; however, the fact that today the number of installed cameras has exceeded human staff available for observation, makes detection and prevention of potential criminal activity on real-time virtually impossible [**16**,**14**].

Despite some current attempts to provide conventional video surveillance systems with intelligence, these systems are still composed of networks of a mixture of CCTV (Closed Circuit Television) and digital cameras under a centralized scheme, where each camera is used as a tool for video acquisition that sends all that video to a central station for storage and partial monitoring by trained personnel [**17**].

In the conventional centralized scheme for video surveillance, the expected growth of video data will double every two years, given that video camera technology is improving every year in terms of resolution, size, and compression rates. This, in turn, introduces new problems such as saturated bandwidth, insufficient and highly error-sensitive real-time monitoring, increase of storage units, and an overall increase in operating surveillance systems [**18**,**19**].

An approach to improve a solution to the aforementioned problems is that conventional systems should evolve to intelligent surveillance systems "on the edge". This type of system, as opposed to conventional systems, pushes intelligence to the edge. Algorithms, embedded at each camera, will analyze retrieved video on real-time and autonomously from its field of vision in search of unwanted events, such that only important information is sent to a central station for monitoring and storage, or simply to alert a human supervisor to take note of the event, thus improving real-time monitoring [**14**,**19**].

A first step towards the autonomous analysis is the recognition of basic human actions. Therefore, *Human Action Recognition* (HAR) from streaming video has

gathered a significant amount (hundreds) of publications since year 2000 [**20**,**21**,**22**,**23**,**24**]. However, most of the proposed methods have been focusing on obtaining increasingly high *recognition rates* without giving significant importance to *complexity* and *computational effort* thus preventing its real-time operation. Then, HAR methods to date do not have enough features to be integrated into the edge of a video surveillance system because they carry out an exhaustive image processing, extracting a high number of features per frame and they use no arithmetic operations (the features are complex to obtain).

Since the aim is contribute in the design of reliable autonomous HAR systems with high recognition rates and the ability to real-time work, in this work, a HAR method that achieves a high performance *Accuracy-Speed-Computational effort* is presented, which might open possibilities for the so-called *"edge processing"*.

1.3 Dissertation Goals

General Goal

The general goal of this dissertation is to design a HAR method based on the natural domain knowledge that is characterized by:

- A *reduced* feature set
- *Real-time* performance
- *Accuracy* competitive with the state of the art methods
- *Pixel resolution beyond Full HD up to 8K*

In order to contribute to the future implementation on *"the edge"* of this type of system

Specific Goals

- Recognize ten o more human actions
- Use two or more publicly available datasets
- Achieve a competitive accuracy without compromising system speed

- Reduce the processing time of feature extraction per frame

1.4 Dissertation Contribution and Organization

The overall contribution of this dissertation is a HAR method based on the *natural domain knowledge* of human actions with attributes such as *precision, speed and simplicity* on a wide range of video resolutions; which might open possibilities for the so-called *"edge processing"* and thus conventional video surveillance cameras can be transformed from *simple tools* for data acquisition and storage into *autonomous intelligent tools* capable of detecting and alerting about potential criminal activity in real-time.

This dissertation is organized in 6 chapters. A brief description of the issue addressed by each chapter is provided next:

- *Chapter 2* presents a review of the most relevant works related to the present investigation
- *Chapter 3* provides fundamental theory that supports the proposed method
- *Chapter 4* describes the proposed method
- *Chapter 5* presents the set of experiments used to evaluate the proposed method
- *Chapter 6* gives conclusions and future directions of this investigation

Chapter 2

2 Related Work

Some of the most relevant HAR methods related to the present investigation are described in this chapter. In section 2.1, a categorization based in the image representation is presented. Then, section 2.2. provides a review of real-time methods. Resolution video in HAR methods is presented in section 2.3 and finally, section 2.4 summarizes the chapter.

2.1 HAR Methods

Currently, the HAR literature has several surveys, some of these publications date from 1995 to 2015 and review papers from 1973 to 2014. Each of these surveys analyzes the works with a different purpose and therefore they use a different taxonomy. For example, some of the most relevant and recent reviews as Moeslum (2006) [**20**] focus on human motion capture and analysis including human model initialization, tracking, pose estimation and action recognition, Poppe (2010) [**21**] focuses on the type of input features used for the classification, Aggarwal and Ryo (2011) [**23**] discuss various approaches at four different levels of activities: gestures, actions, interactions, and group activities, Ke *et al.* (2013) [**24**] covers the three representation levels of HAR, from core technology (low-level), the human activity

recognition systems (middle-level) and their relevant applications (high-level), and Afsar *et al.* (2015) [**25**] classify works into three main subjects, detection techniques, datasets and applications.

Based on the image representation used to extract the features that will be used to recognize the action, previous methods in this dissertation will be organized into two large groups, those that use a *holistic representation* and those that use a *part-based representation*. Methods based on a holistic representation use the whole ROI information to characterize the action. ROIs, which contain the whole human silhouette, are used in the following papers. In [**26**] Blank *et al.* represent human actions as three-dimensional objects generated by grouping of silhouettes in volumes on the space-time domain. In [**27**] Guo *et al.* modeled an action as a temporal sequence of deformed centroid-centered silhouettes. In [**28**] Chen models the action as a sequence of parameters from star figures represented as Gaussian Mixture Models (GMM), where a star figure is bounded by the smallest convex polygon containing the human silhouette. Other works have also used the whole ROIs, but on raw images instead of silhouettes. In [**29**] Schindler and Van Gool recognize the human action of short sequences of video (snippets) using shape information (local edges) and movement (optical flow) on raw images. Derpanis *et al.* [**30**] generate three-dimensional volumes by grouping whole ROIs on a space-time map and measuring energy through derivatives and widely tuned three-dimensional Gaussian filters. Instead of using the whole image contained by the ROI, other methods are based on rectangular image patches for extraction of information to characterize human action (part-based representation). In the following methods, the ROI is divided in an equal-sized grid. Jimenez *et al.* [**31**] propose a multi-scale HAR descriptor based on a *Pyramid of Accumulated Histograms of Optical Flow* (PaHOF). Optical flow between two consecutive frames is represented as histograms of orientation vs. magnitude accumulated over time and computed for each cell on a grid of non-overlapping regions. In [**32**], Ikizler *et al.* proposed a pose descriptor for HAR, called *histogram of oriented rectangles* (HOR), where the ROI is divided on an equal-sized grid and the HOR is computed within each cell. In [**33**], Baysal and Duygulu use a part-based representation for HAR, where information of speed and

direction of movement are used along with a pose representation based on a collection of line-pairs adjusted to the contour of the human figure.

2.1.1 Discussion

For the case of methods based on the holistic representation there is an extraction of features from the complete ROI where the whole ROI image is analyzed and the length of the description vector is usually fixed. General methods based on the holistic description are susceptible to noise, occlusion and view-point variation so that these methods work properly on controlled environments.

In the case of part-based methods, there is a feature extraction process in a patch-by-patch basis where each set of features is considered independent and of equal importance. In general, these methods tend to be more robust to noise, occlusion and in some cases invariant to rigid transformations; however, these methods present a significant disadvantage which is that the size of the description vector is usually very large and variant according to the number of patches used.

Holistic and part-based representations have different strengths and weaknesses. As a consequence, some researchers have used a mixed representation (holistic + part-based). For example, Wang *et al.* [**34**] extend the *probabilistic Latent Semantic Analysis* (pLSA) model to HAR by using bag-of-words. Each frame is encoded using a descriptor *Pyramid of Histogram of Oriented Gradient* (PHOG) which encodes the human figure to multiple degrees of detail according to different pyramid levels. In [**35**], Bregonzio *et al.* represent a human action as clouds of points at different time scales where shape and movement features are extracted from two regions: the ROI containing the segmented object and the areas generated by the clouds of points at different scales. Minhas *et al.* [**36**] present a method for incrementally HAR, which adaptively extracts PHOG features from the full body and three sub-regions based on a strategy of tracking that uses form and appearance.

The methods, such as those cited above and part-based, use a joint representation of the image and a description vector of large size which turns out to be a disadvantage in practical systems. As a consequence, in this work a joint

(holistic + part-based) representation of image along with a description vector of small size are used.

2.2 Real-Time HAR Methods

As it was mentioned previously, the aim is to collaborate in the design of HAR solutions suitable for real-time scenarios "on the edge". For this reason, a review of the state of the art to those methods that reporting real-time operation with any timing evaluation on publicly available datasets will be given next.

In this section, the real-time methods will be categorized in base to the image representation described above. The next methods use a *holistic representation.*

One of the first studies that report a real-time HAR method was presented by Bobick and Davis in [**37**]. This method uses moment-based features to represent the action. The features (7 Hu moments) are extracted from two types of images, motion-energy image (MEI) and motion-history image (MHI) to form temporal templates and then, these templates are matched against stored models of known movements to recognize the action. Hernández *et al.* [**38**] use a visual tracking of subject based in bounding box (BB) and obtain features as height, width and position of BB for each frame. Then, evolution in time of these features feed a *SVM* classifier to recognize de action. Cheema *et al.* and Chaaraoui *et al.* in [**39**,**40**,**41**] use features based on the points of the silhouette contour. In [**39**,**40**] authors compute the Euclidean distance from each silhouette contour point to the silhouette centroid and then use these data to find the most characteristic poses (*key poses*). Subsequently, in [**39**] weights are assigned to the key poses and a *weighted voting scheme* is applied to recognize the action. Otherwise, in [**40**] temporal information is added by generating sequences of key poses, and finally *Dynamic Time Warping* (DTW) is used to classify sequences. A new feature based on silhouette contour and a radial scheme is used in [**41**], in this method Chaaraoui *et al.* divide the silhouette's contour in S radial bins and all contour points are assigned to the corresponding radial bin, then a summary representation is obtained for each bin, after clustering is used to identify per-view key poses (*bag of key* poses), and finally the recognition is performed using DTW. Natarajan and Nevatia in [**42**] presents a top-down

approach to simultaneously track and recognize articulated full-body human motion using learned action models based on the *Hierarchical Variable Transition Hidden Markov Model* (HVT-HMM). This model is introduced in their work and consists of three layers that model composite actions, primitive actions and poses. On the other hand, the next real-time methods use a *part-based representation*. In [**43**] Guo a method based on "bags of words" (BOW) representation and the model probabilistic Latent Semantic Analysis (pLSA) is presented. Patches of interest are used to represent and describe action sequences. In [**44**], Meng *et al*. use the pixels of MHI images obtained from human action videos as feature vectors to train a SVM multiclass made up by six binary SVMs (one per class). Chakraborty *et al*. in [**45**] first represent the human body as a stick figure based on three lines (the two legs and the trunk) and extract features based on the angles between these lines and the vertical axis. Then, standard deviation of the angular features from each video sequence and the mean of these standard deviations are used to characterize the action. A method for action recognition based on a set of features obtained from some moments of frame differences (region of movement) is presented by Sadek in [**46**]. Features are related to tracking of the centroid (center of movement, intensity of movement and mean absolute deviation from the center of motion) and classification is performed using *SVMs*. Finally, the next real-time methods use a Holistic + part-based representation. Kalhor *et al.* in [**47**] it describes a method for action recognition using a descriptor based on appearance and movement of the human silhouette with a classifier called *Time Delay Input Radial Basis Function Network* (TDIRBF). Movement describing features are based on the centroid and BB corners and shape describing features are based on radial histograms of 18 bins on a 2 x 2 grid. Shape and movement features are used by Hernández *et al*. in [**48**]. They find the BB and divide the silhouette in horizontal regions of equal height, then compute some measures as variance, average, covariance etc. of the characteristic parameters of each region and finally these features are fed to a SVMs system to classify the action.

In particular, the method presented in this work is similar to method in [**48**]. However, the proposed method differs to others in the following points:

- It represents the human silhouette as a set of *three* rectangular boxes
- The rectangular boxes are of different height and width
- It uses a hierarchical system of classifiers (a perceptron and three multi-class SVMs)
- It uses a combination of very computationally simple techniques at its different stages

As a consequence, the method proposed in this work achieves better results in terms of speed and accuracy; even as speed is reported in [**48**] only for its "tracking stage".

Finally, Table 2.1 resumes relevant characteristics of HAR methods that reported real-time performance. In the first column, the reference of the method is provided. Second column gives the publication year. Third column shows the input type; images (I), silhouettes (S) and binary human contours (C). On the fourth column, the datasets used to assess each method are given. Fifth column succinctly gives the features used to recognize the action. Finally, columns six and seven indicate if a method reports a value for accuracy (%) and processing time (fps) respectively.

2.3 Resolution Video in HAR methods

The rapid evolution of video surveillance technology based on computer vision is consequence of the increasing demands for security and safety by society. Development started with the introduction of pixel surveillance camera, then megapixels camera, after HD (High Definition) and Full HD cameras and today UHD (Ultra High Definition) surveillance camera.

The growing need for higher image resolution in video surveillance is a consequence of the need to improve the quality of footage. Higher video definition provides clearer images and crisper videos to identify and monitor actions of criminals readily and generate images that can be used as irrefutable evidence against criminals in a court law.

Table 2.1. Summary of HAR methods showing real-time performance.

Method	Year	Input	Datasets	Features	%	fps
[37]	2001	I	Their own	They compute 7 Hu moments of MHI and MEI images to build temporal templates	✓	—
[44]	2008	I	KTH and their own	They use the pixels of MHI images as features	✓	—
[42]	2008	S	Weizmann, KTH and their own	They use learned action models based on HVT-HMM	✓	✓
[45] [a]	2009	C	KTH	They extract angular features of body parts using a skeleton technique	✓	✓
[43]	2010	I	Weizmann, KTH and their own	They extracts patches of interest	✓	✓
[39]	2011	S	Weizmann and MuHAVi	They compute euclidean distance from each silhouette contour point to the silhouette centroid to find key poses	✓	✓
[38]	2011	I	weizmann	They use shape features of BB and their evolution in time	✓	✓
[49] [a]	2012	I	weizmann	They use shape features and moments computed from images obtained by frame difference	✓	✓
[47]	2014	S	UIUC	They use movement features based on the centroid and BB corners and shape features based on radial histograms	✓	✓
[40]	2013	S	Weizmann, IXMAS and MuHAVi	They use features based in the silhouette contour to find key poses	✓	✓
[41]	2014	S	Weizmann, MuHAVi, and their own	They propose features based on silhouette contour and a radial scheme	✓	✓
[48]	2014	S	Weizmann, KTH, IXMAS and UIUC	They use measures as variance, average and covariance of different parameters taken from silhouette's horizontal regions	✓	✓
[50]. [a]	2014	S	Weizmann and KTH	They use shape-based pose features extracted from the surrounding regions (negative space)	✓	✓

[a] real-time for frame rate slower than 25 fps

With the ever increasing of image quality, video surveillance benefits by easing criminal actions detection; but on the other hand, higher quality image generates higher computational and communication load and hence the need to propose new and better solutions for High definition video analytics. This is one of the main objectives of this work.

The research topic of Human Action Recognition (HAR) from streaming video has gathered a great number of publications. However, very few methods reported in those publications are applicable for real-time processing and a very smaller percentage of the HAR methods performed in real-time used HD videos. The great majority of the research is focused on obtaining high recognition rates without giving relevance to computational effort, speed and constant increase in image quality. Table 2.2 summarizes the real-time HAR methods and the resolution video on which was evaluated.

Table 2.2. Review of real-time HAR methods and the video resolution used.

Method	**Year**	**Input**	**Resolution video**		
			Low	*Medium*	*High*
Bobick and Davis [**37**]	2001	Grey scale images	✓	—	—
Meng *et al.* [**44**]	2008	Grey scale images	✓	—	—
Natarajan and Nevatia [**42**]	2008	silhouettes	✓	—	—
Guo [**43**]	2010	Images	✓	—	—
Cheema *et al.* [**39**]	2011	silhouettes	✓	✓	—
Hernández *et al.* [**38**]	2011	images	✓	—	—
Sadek *et al.* [**49**] [a]	2012	images	✓	—	—
Kalhor *et al.* [**47**]	2014	silhouettes	—	—	✓
Chaaraoui *et al.* [**40**]	2013	silhouettes	✓	✓	—
Chaaraoui and Flores-Revuelta [**41**]	2014	silhouettes	✓	✓	—
Hernández *et al.* [**48**]	2014	silhouettes	✓	✓	✓
Rahman *et al.* [**50**] [a]	2014	silhouettes	✓	—	—

[a] real-time for frame rate slower than 25 fps.

From the methods showed in Table 2.2, it is observed only two [**47**,**48**] have been tested on high resolution videos showing real-time performance. Drawn from this analysis and motivated to give new solutions with real-time performance for full HD

videos, a real-time HAR method that achieves a high Accuracy-Speed performance on videos up to 8K UHD is presented.

2.4 Summary

In this chapter, the most relevant works related to present investigation were presented. First, according to representation used to extract the features; previous methods were divided into two groups, those using a *holistic representation* and those using a *part-based representation*. The methods using a holistic representation take the whole ROI information to characterize the action while the methods using a part-based representation take rectangular image patches to obtain information to characterize human action. Then, a discussion about strengths and weaknesses of each representation let us introduce a third group of works which shows better attributes since use a mixed representation (holistic + part-based). Because the objective of this work contributes to design of HAR solutions suitable for real-time scenarios "on the edge", a review of the state of the art methods reported with real-time operation giving any timing evaluation were presented and finally, a review of video resolution used by them was shown too.

Chapter 3

3 Theory Fundamentals

In this chapter relevant concepts that will be used in subsequent chapters are presented, this in order to make this document self-contained. This chapter is divided in three parts. The first part provides basic terminology and background information about Human Action Recognition. The second part gives a brief explanation about the object segmentation method used for this work, and finally the third part describes the classifiers used in the action classification stage.

3.1 Human Action Recognition (HAR)

Human action recognition (HAR) based on video sequences is a topic of great interest in computer vision research. The goal of HAR consist in automatically analyze and deduce the action or actions executed by one or more persons from a video or images scenes [**51**].

3.1.1 Motion Hierarchy

Currently there are a great variety of hierarchies related to the categorization of the movement; terms like action and activity are used interchangeably by different

authors. In order to use an unified motion categorization in this work, the motion hierarchy proposed by Moeslum *et al.* [**20**] will be adopted. This motion hierarchy (Figure 3.1) has three levels and the complexity of each level is higher than the previous.

An *activity* is the highest hierarchy element and it is a combination of its consecutive *actions*. An action is a conjunction of primitive actions and an *action primitive* is a movement of a human limb. For example, "javelin throw" is an activity which combines "running" "jumping" and "throwing " actions, and "right-arm-up", "right-arm-forward" are actions primitives of "throwing" action. This work will focus on the recognition of human movements at level of "action".

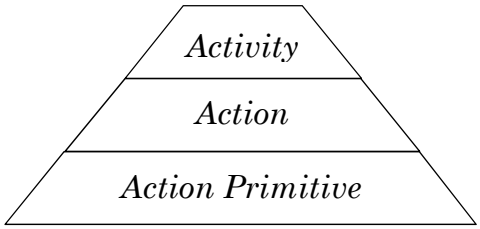


Figure 3.1. Diagram of motion hierarchy.

3.1.2 HAR Processing Stages

In general, a human action recognition system at low-level can be represented by three main processing stages (Figure 3.2): object segmentation, feature extraction and representation, and action classification [**24**]. In the first stage, the target object is segmented from each frame in the video sequence. Then, characteristics of the segmented object such as shape, size, colors, poses and body motions are obtained and formally represented in form of extractable features. Finally, classification algorithms based on the extracted features are used to recognize different human actions.

Figure 3.2. Main processing stages of HAR method.

3.2 HAR Evaluation

In general, most of the studies in the HAR area evaluate and compare their methods performance based on the accuracy obtained on one or more datasets using some kind of Cross-Validation (CV). Therefore, this section describes the most used evaluation methodologies based in CV and the concept of confusion matrix.

3.2.1 Cross-Validation (CV)

Cross-Validation (CV) is a validation strategy model whose main attribute lies in the universality of the data splitting [**52**]. It only assumes that data used for evaluating the performance of the algorithm are uniformly distributed, and training and validation samples are independent. Figure 3.3 shows the hierarchy of some common types of CV. There are two main types of CV in the first level: exhaustive and non-exhaustive. Exhaustive strategies validate the model using all possible ways of partitioning the original dataset into training and validation sets, and non-exhaustive strategies are considered an approximation of exhaustive methods because they use only some ways of partitioning the original dataset.

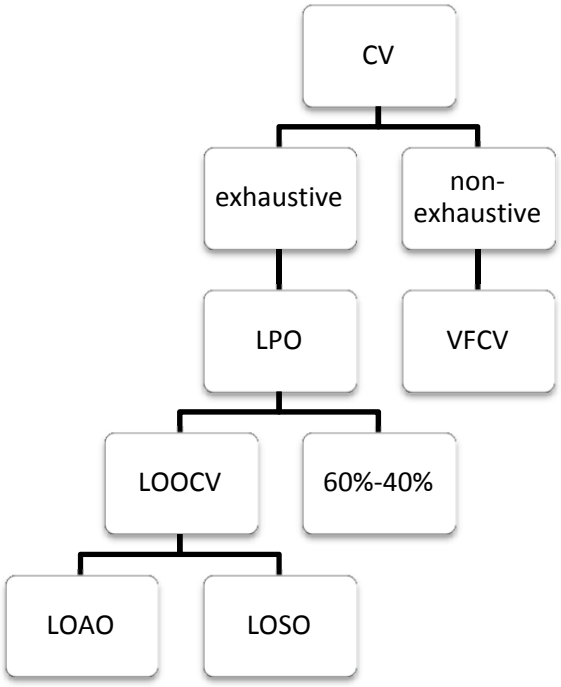


Figure 3.3. Hierarchy of some common types of Cross-Validation.

In the second level, inside non-exhaustive methods *V-fold cross-validation* (VFCV) is found. VFCV partitions data in V folds of approximately equal size *n*/V where *n* is the total of data. Each fold is used once as testing set and the remaining V-1 folds as training set. This process is repeated V times and the results of all iterations are averaged to obtain the final result.

At the same second level, but as exhaustive methods *Leave-P-Out-Cross-Validation* (LPO) is found. LPO uses n_t training samples; where $n_t = n - p$ given n as the total of data and p the number of validation samples with $p \in \{1, \dots, n-1\}$. Each time a possible subset of p data is successively *"left out"* of the sample and used for validation. This procedure is repeated $\binom{n}{p}$ times and the result is reported as the average of these runs. Thus, training is achieved using, basically, all samples, and at the same time independence between training and test sets is maintained.

Inside LPO, two particular cases used in this work will be described. *Leave one out cross validation* (LOOCV) and 60%-40%. LOOCV is the most classical exhaustive CV procedure and it is a particular case of LPO with $n_t = n - 1$. Then, each data point is successively *"left out"* from the sample and used for validation while the remaining $n - 1$ points are used for training. This procedure is repeated n times and the partial results are averaged. 60%-40% is another particular case of LPO with $n_t = n - 0.4n$. It uses $0.6n$ samples for training and $0.4n$ for validation. The process is repeated $\binom{n}{0.4n}$ times and the experimental results are reported as the average of the partial outcomes.

In the particular case of HAR literature two types of strategies are used, *Leave-One-Actor-Out cross-validation* (LOAO) and *Leave-One-Sequence-Out cross validation* (LOSO) depending on which element; actor or sequence is selected to conduct experiments.

When one actor is selected as the element to conduct experiments using LOOCV, this strategy is known as LOAO, which verifies actor-variance. For a dataset with k different actors performing a set of actions, all action video sequences from one actor are used to test the algorithm, while the sequences of the remaining $k - 1$ actors are used for training. The process is repeated for k times and the experimental results are reported as the average of the outcomes from those k runs.

When one sequence is used as the element to conduct experiments using LOOCV, this strategy is known as LOSO. For a dataset with n action video sequences, the LOSO "left out" only the testing action sequence and the remaining *n-1* are used for training.

3.2.2 Confusion Matrix

A confusion matrix $M_{k\,x\,k}$ is a mathematical tool that allows us to visualize the results obtained by a classifier of k classes with respect to some validation dataset [**53**]. One matrix dimension corresponds to the true class and the other dimension is the inferred class by the algorithm. The number of true classifications of each class is shown on matrix diagonal, off-diagonal elements appear when classifications errors happening. Table 3.1 shows an example of confusion matrix augmented by its row and column total for a standard two-class problem. Suppose N samples have been classified as either X or $\bar{X}$ ("not X"). In truth, there are $a + c$ samples are X and $b + d$ samples are $\bar{X}$; the algorithm "believes" that there are $a + b$ samples that are X and $c + d$ samples $\bar{X}$. Then, there are $a + d$ correct, and $b + c$ erroneous.

Table 3.1. Confusion Matrix for standard two-class problem.

	X	$\bar{X}$	total
X	a	b	$a+b$
$\bar{X}$	c	d	$c+d$
total	$a+c$	$b+d$	N

Some of common values computed from a confusion matrix are showed in Table 3.2.

Table 3.2. Evaluation measures obtained from confusion matrix.

Measure	Value
Sensitivity	$a/(a+c)$
Specificity	$d/(b+d)$
Positive predictive value	$a/(a+b)$
Negative predictive value	$d/(c+d)$
Odds ratio	$(ad)/(cb)$
Correct classification rate	$(a+d)/N$

From these values, *Correct Classification Rate* (CCR), also known as a*ccuracy* is the most used metric because it represents the overall efficiency for all classes.

3.3 Object Segmentation

Almost all HAR systems start with the object segmentation and therefore it is crucial to the subsequent stages. The object segmentation consists in separating the object of interest (person) from the rest of the image. Different object segmentation methods have been used in HAR area; for example background subtraction, statistical methods, temporal differencing, and optical flow. However, based on the datasets characteristics and the simplicity and efficiency of the background subtraction technic, it was used in this work.

3.3.1 Background Subtraction

Background subtraction methods (BS) are techniques to detect moving objects in videos from fixed cameras [**54**]. These techniques despite their differences are based on the assumption that the observed video sequence I is carried out using a fixed background B in front of which moving objects are observed and that the moving objects at time t have a color distribution different from the one observed in B. Then, the principle of BS methods is summarized in the next formula:

$$\chi_t(s) = \begin{cases} 1 & if\ d(I_{s,t}, B_s) > \tau \\ 0 & otherwise, \end{cases} \tag{3.1}$$

Where χ_t is the motion label field at time t, d is a distance between $I_{s,t}$ the video frame at time t at pixel s and B_s the background at pixel s, and τ is a threshold. The difference between BS methods is how B is modeled and which distance metric d is used. This work use the most basic way to model the background. In this method, B is a grayscale image taken in absence of moving objects, d is the distance metric defined in equation (3.2) and foreground pixels are detected by Otsu thresholding method.

$$d = \left| I_{s,t} - B_{s,t} \right| \tag{3.2}$$

3.3.2 Otsu Thresholding Method

The Otsu method [**55**] is a clustering-based thresholding method which is completely performed on the histogram of a frame, and it is optimum in the sense that it estimates a *threshold* τ that minimizes the variance within a cluster (*intra-class* variance) and at the same time it maximizes the variance between the two clusters (*inter-class variance*).

Consider L distinct intensity levels in a frame $[0, 1, \dots, i, \dots, L-1]$, a *histogram* with entries $[p_0, p_1, \dots, p_i, \dots, p_{L-1}]$, and a threshold τ to segment a frame into two regions; the silhouette or *foreground* (darker) consisting of pixels with intensity values in the range $[0, \tau]$ and the *background* (brighter) consisting of pixels with values in the range $[\tau + 1, L-1]$. The inter-class variance σ_I^2 is a measure of the separability between classes (foreground and background) and it is defined as

$$\sigma_I^2 = P_F\,(m_F - m_G)^2 + P_B\,(m_B - m_G)^2 \tag{3.3}$$

where $m_G = \sum_{i=0}^{L-1} i\,p_i$ is the global mean of the entire frame, $P_F = \sum_{i=0}^{\tau} p_i$ is the probability that a pixel is assigned to the foreground, $m_F = \frac{1}{P_F} \sum_{i=0}^{\tau} i\,p_i$ is the mean intensity value of pixels belonging to the foreground, and similarly $P_B = \sum_{i=\tau+1}^{L-1} p_i$ and $m_B = \frac{1}{P_B} \sum_{i=\tau+1}^{L-1} i\,p_i$ are the probability and the mean intensity values of pixels on the background.

In the Otsu method, a search for the threshold is performed as the value of τ is varied until σ_I^2 is maximized. Once σ_I^2 is obtained, the grayscale frame is binarized according to

$$b(r,c) = \begin{cases} 1 & if \quad f(r,c) > \tau \\ 0 & if \quad f(r,c) \leq \tau \end{cases} \tag{3.4}$$

3.4 Classification

As it was mentioned previously, the last stage in a HAR system is the action classification. In this stage, most of the methods assign an action label to each frame or sequence. Some classification algorithms such as DTW, generative models and discriminative models have been used to recognize human actions. However, in this work a hierarchical system of classifiers has been proposed. This system uses two types of classifiers, perceptron and support vector machine. Hence, these two types of classifiers will be defined in the following sections.

3.4.1 Perceptron

The perceptron is the simplest form of a neural network used for the classification of patterns said to be linearly separable [**56**]. Its model is shown in Figure 3.4. It was made in analogy to a human nerve cell and consists of a set of inputs $x_1, x_2, \dots, x_m$, a set of synapses, each of which is characterized by a weight $w_1, w_2, \dots, w_m$, an adder (Σ) for summing the input signals weighted by the respective synaptic strengths, an activation function ($\varphi(\cdot)$) for limiting the amplitude of the output, and an externally applied bias b for increasing or lowering the net input of the activation function.

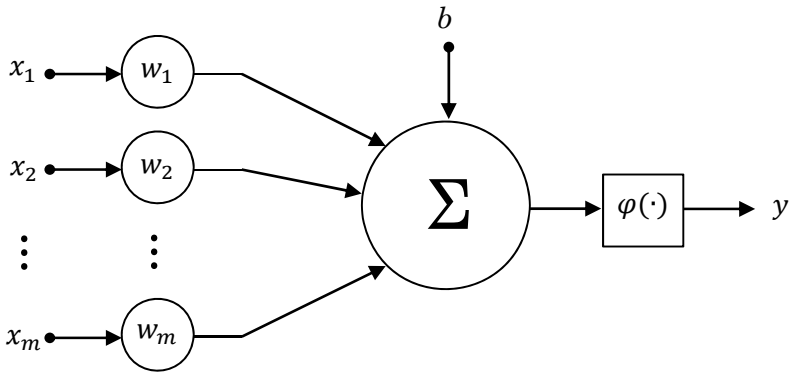


Figure 3.4. Model of a basic perceptron.

The perceptron behavior for the case of two linearly separable classes is shown in Figure 3.5, it is a map of the decision regions in the l-dimensional feature space

$\boldsymbol{x} \in \boldsymbol{R}^l$ where l=2. The map is separated into two regions $\mathcal{C}_1$ and $\mathcal{C}_2$ by a linear hyper-plane,

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b} = \boldsymbol{0} \tag{3.5}$$

where $\boldsymbol{w} = [w_1, w_2]^T$ is the *synaptic weight vector* and b is the *bias*; and these are parameters to be learned in this model, called *perceptron*. The hyper-plane is a straight line on the two-dimensional space with the slope depending on coefficients w_1 and w_2, and with the bias b controlling the y-intercept.

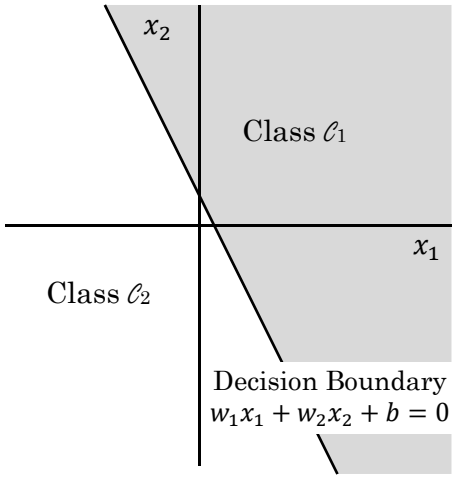


Figure 3.5. Region of decision for the case of two linearly separable classes.

The optimal hyper-plane is learned from a training set of m observations $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^m$ coming from two linearly separable classes $\mathcal{C}_1$ and $\mathcal{C}_2$ where $d_i = 1 \; \forall \, f(\boldsymbol{x_i}) > 0$, and $d_i = -1 \; \forall \, f(\boldsymbol{x_i}) < 0$. The perceptron converges by iterative adaptation of the weight vector $\boldsymbol{w}$ according to the following adaptation rule at iteration n [**57**],

$$\Delta \boldsymbol{w}(n) = \eta \, e(n) \, \boldsymbol{x}(n) \tag{3.6}$$

where η is the learning rate and $\mathrm{e}(n) = d(n) - y(n)$ is the error between the desired perceptron response $d(n)$ and the actual perceptron response $y(n) = sgn[f(\boldsymbol{x}(n))]$ at iteration n. At each step, $\boldsymbol{x}(n)$ and $d(n)$ are randomly chosen from the training set.

3.4.2 Support Vector Machine (SVM)

A linear support vector machine (SVM) is a binary learning machine that builds an optimal linear hyperplane from a training set in such a way that the margin of separation between feature vectors of two different classes is maximized [**56**]. Figure 3.6 shows the optimal hyperplane obtained from a SVM for two classes.

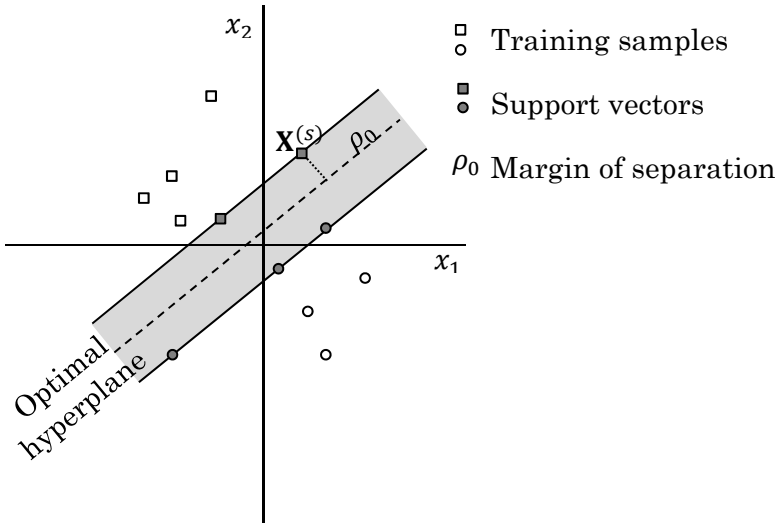


Figure 3.6. Optimal hyperplane for two linearly separable classes.

Given the training sample $\mathcal{T} = \{(\boldsymbol{x}_i,\ d_i)$ with $d_i \in \{-1,1\}\}_{i=1}^{m}$, it wants to find the optimal hyperplane $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b = 0$ that separates; the patterns with target output $d_i = 1$ from those with $d_i = -1$. Then, once again the goal is to find optimum values of the parameters $\boldsymbol{w}$ and b such that they satisfy the constraints

$$arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \qquad subject\ to \qquad d_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 \qquad for \qquad i = 1, \dots, m \tag{3.7}$$

This constrained optimization problem is called the *primal problem*. The primal problem can be stated equivalently by its *dual representation* form as follows:

Given the training sample $\mathcal{T} = \{(\boldsymbol{x}_i,\ d_i)\}_{i=1}^{m}$, it wants to find the Lagrange multipliers $\{\lambda_i\}_{i=1}^{m}$ that satisfy

$$arg\max_{\lambda}\left(\sum_{i=1}^{m}\lambda_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_i\lambda_j d_i d_j \boldsymbol{x}_i{}^T\boldsymbol{x}_j\right)\quad subject\ \ to\quad \sum_{i=1}^{m}\lambda_i d_i=0\ and\ \lambda_i\geq 0 \tag{3.8}$$

Once λ_i are computed according (3.8), it has the optimum Lagrange multipliers $\lambda_{0,i}$ and the optimal linear hyper-plane is obtained through optimum weight vector

$$\boldsymbol{w_0}=\sum_{i=1}^{N_S}\lambda_{0,i}d_i\boldsymbol{x}_i \tag{3.9}$$

where N_S is the number of support vectors. Therefore, the weight vector $\boldsymbol{w_0}$ is a linear combination of the set of *support vectors* which are the feature vectors associated to non-zero Lagrange multipliers $\lambda_{0,i}$. The optimum bias $b_0$is computed as an average value of all conditions $\lambda_{0,i}[d_i f(\boldsymbol{x_i})-1]=0; i=1,\dots,m$.

3.4.2.1 SVM for Non-Separable Patterns

As it was mentioned above, the goal of a SVM is to build an optimal hyperplane able to completely divide a set of input patterns in two different classes. However, a total separation of the patterns is not always possible, as shown in Figure 3.7, where a pattern is positioned on the wrong side of the decision surface.

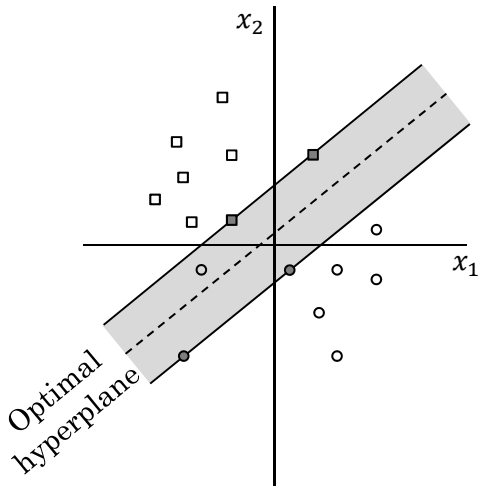


Figure 3.7. Example of non-separable patterns.

In order to allow the classification of non-separable patterns with some misclassification, the SVM model incorporates a C parameter that controls the tradeoff between complexity of the machine and the number of non-separable points.

The optimization problem becomes

$$arg\min_{\boldsymbol{w},\boldsymbol{\varepsilon}}\left[\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\xi_i\right] subject\ to\ d_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) \geq 1 - \xi_i\ and\ \xi_i > 0; i = 1, \ldots, m. \tag{3.10}$$

where ξ_i are known as *slack variables* and they measure the deviation of a data point from the ideal condition of pattern distinguishability.

3.4.2.2 Multi-Class SVM Classifier

The SVM classifier can be extended to a multiclass problem with a set of output states $\mathcal{y} = \{1,2,\ldots,M\}$ using the set of discriminant functions $f_y(\boldsymbol{x})$, $y \in \mathcal{y}$ and classification rule

$$label = arg\max f_y(\boldsymbol{x}),\ y \in \mathcal{y} \tag{3.11}$$

There are two main ways to design the multiclass SVM classifier based in equation (**3.11**). The decomposition based methods and the multiclass SVM formulation solved in a single optimization.

Between decomposition based methods, *One-Against-All* and *One-Against-One methods* are found. In the *One-Against-All* decomposition, the multi-class classifier is posed as a set of M discriminant functions $\{f_i(\boldsymbol{x}) = 0\}_{i=1}^{M}$ where a hyper-plane is learned to separate class C_i from the rest of the classes; and $f_i(\boldsymbol{x})$ is used as a discriminant function which satisfies $f_i(\boldsymbol{x}) > f_j(\boldsymbol{x})\ \forall\ i \neq j\ and\ \boldsymbol{x} \in \mathsf{C}_i$. Given a feature vector $\boldsymbol{x}$**,** a label is assigned according to

$$label = arg\max\nolimits_{i=\{1,\ldots,M\}} f_i(\boldsymbol{x}). \tag{3.12}$$

In the *One-Against-One* implementation, the goal is to train a multi-class SVM based on the majority voting rule. Given M different classes $\{C_i\}_{i=1}^{M}$, the classification

problem is posed as learning a set binary discriminant functions $\{f_i(\boldsymbol{x}) = 0\}_{i=1}^{\frac{M(M-1)}{2}}$ where $\frac{M(M-1)}{2}$ is number of combinations of two different classes $\{C_i, C_j \mid i = 1, \dots, M-1; j = i+1, \dots, M\}$, and a hyper-plane is learned to classify an input feature vector $\boldsymbol{x}$ into two corresponding classes C_i and C_j. Let $\boldsymbol{v}$ be the vector with entry v_i corresponding to the total number of votes when $\boldsymbol{x}$ is classified into class C_i, then the multi-class SVM assigns a label to $\boldsymbol{x}$ according to

$$label = arg\max_{i=\{1,\dots,M\}} v_i. \tag{3.13}$$

Some shortcomings of previous multi-class methods are the generation of the training set for each individual SVM classifiers and that there are unclassified training samples.

In the multiclass SVM formulation solved in a single optimization, the *multi-class Bias SVM* (BSVM) classifier is found. It does not require the generation of several training sets or the learning of multiple binary SVM classifiers [**58**]. Given a set of m training observations $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^{m}$ with class index $d_i \in \{1, \dots, M\}$, the parameters for the *multi-class BSVM* are solved according to the following minimization problem

$$\begin{aligned} \{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\xi}\} = \arg\min_{\boldsymbol{W},\boldsymbol{b},\boldsymbol{\xi}} \frac{1}{2}[\|\boldsymbol{W}\|_F^2 + \|\boldsymbol{b}\|_2^2 + C\|\boldsymbol{\xi}\|_F^2] \quad subject\ to \\ \boldsymbol{w}_{d_i}^T \boldsymbol{x}_i + b_i - \left(\boldsymbol{w}_j^T \boldsymbol{x}_i + b_j\right) \geq 1 - \xi_{d_i}^j, \xi_{d_i}^j > 0, \quad j \neq d_i. \end{aligned} \tag{3.14}$$

where $\|\boldsymbol{W}\|_F$ is the Frobenius norm of matrix $\boldsymbol{W}$, $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_M]$ is a matrix with M columns and each of these column vectors corresponds to the weight vector $\boldsymbol{w}_i$ for classifier $f_i(\boldsymbol{x})$, $\boldsymbol{b} = [b_1, b_2, \dots, b_M]^T$ is the vector with entry b_i corresponding to the bias for classifier $f_i(\boldsymbol{x})$, and $\boldsymbol{\xi}$ is a matrix of slack variables with each column vector $\boldsymbol{\xi}^i$ corresponding to class C_i.

3.5 Summary

This chapter presented the fundamental theory that supports the proposed HAR method in this dissertation. First, relevant theory related to human action recognition is mentioned. The chapter starts describing the concept of human action recognition. Then, the categorization of motion used in this work; activity, action and action primitive is established. Next, the main stages of the HAR process; object segmentation, feature extraction and action classification are presented. Afterward, the HAR evaluation process based in CV and confusion matrix is described. Subsequently, object segmentation technique based in BS and Otsu thresholding is presented. Finally, the two types of classifiers used in classification stage are exposed.

Chapter 4

4 Proposed HAR Method

The human action recognition based on video sequence has become of great interest in research, security agencies, and industry. This is mainly due to the relevance of its potential applications. However, for HAR methods to be truly useful in most of these applications, it is essential they show real-time performance and in some applications such as video surveillance they require fewer hardware resources in order to be implemented at the edge. These attributes are difficult to achieve for most of the proposed methods because the features that they use are highly elaborated, abundant in number, and complex to obtain. Therefore, this chapter presents a HAR method that incorporates the natural domain knowledge of the problem to action recognition in order to provide a solution with significantly reduced features. In section 4.1 a general description of method is given. The evolution of method is described in section 4.2, and sections 4.3 to 4.7 detail each one of the method stages.

4.1 Overview of the Method

The workflow of the proposed HAR method is shown in Figure 4.1. The process consists of two stages: (a) training and (b) testing. The training stage receives as

input a set of video sequences of different actions; then generates a set of feature vectors and their corresponding action labels; which will be used in the learning process to configure a system of classifiers. The testing stage takes as input an action video sequence, obtains its feature vector and assigns an appropriate action label to the input. In the testing stage, most of the sub-processes are the same as those in the training stage (preprocessing, BB tracking, BB representation, features extraction and feature vector) except the Classifier Model; it is re-used to predict the class of the input video. The sub-processes common to training and testing stages are described next. During the *preprocessing* sub-process, binary silhouettes are obtained from full action videos using simple segmentations technics and then these silhouette sequences are divided in sub-sequences of *n* frames, known as *snippets*. Next, for each snippet frame, a *tracker* locates the smallest rectangle that encapsulates the human silhouette called *Bounding Box* (BB). Once the BB is defined, the human silhouette is represented using this BB rectangle, and two smaller rectangles KB and FB (Knee Box and Feet Box), contained within the original BB. Later, two types of features are extracted, *local features* and *global features*. *Local features* describe the morphology of a silhouette and are computed for each frame; and *global features* describe how movement unfolds and are computed for each snippet. Finally, global features are concatenated into a single *feature vector*.

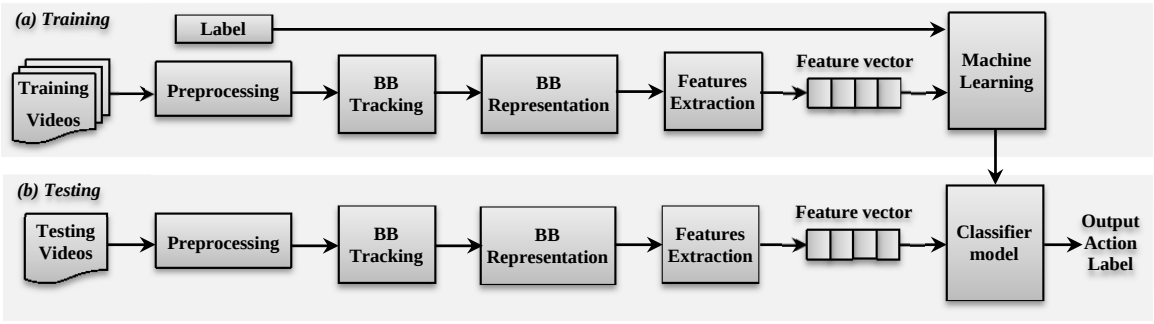


Figure 4.1. The workflow of the proposed HAR method.

4.2 Method Evolution

The HAR method described in section 4.1 was initially designed to recognize a set of 4 actions *{wave1, jack, walk, and run}* using 4 features (*Method 1*) and later, it was complemented to recognize a set of 10 actions *{wave1, wave2, bend, pjump, jack, walk, run, side, skip, and jump}* by using 6 features (*Method 2*). Both methods have the same framework (Figure 4.1). However, they differ slightly in the three sub-processes showed in detail in Figure 4.2: 1) BB representation, 2) Features extraction and 3) classifier model. Regarding the BB representation sub-process, the method 1 uses two (BB and KB) of the three rectangular boxes required by the method 2 to model human body. In the case of features extraction sub-process, the method 1 extracts four of the six local features required by the method 2. Three extracted from BB (width, centroid abscissa, upper edge coordinate) and one from KB (width), and five of the seven global features. Four extracted from BB (maximum width, range width, maximum horizontal displacement and maximum scrolling) and one from KB (maximum width). Finally, with respect to the classifier model, the method 1 uses three of four classifiers required by the method 2 (one perceptron and two SVMs), but SVMs are binary classifiers in this case. It shows the proposed method in this work is flexible and with slight modifications can be adapted to recognize a greater number of actions.

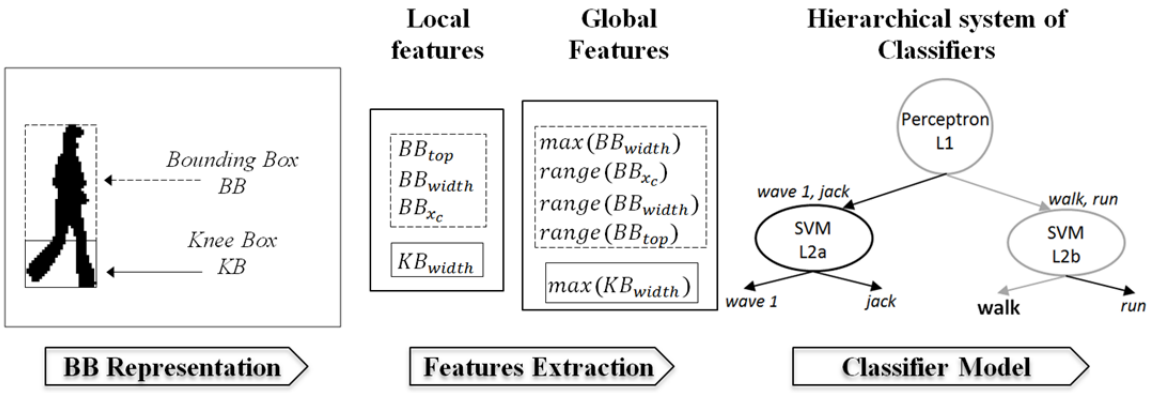


Figure 4.2. Sub-processes characteristic of the HAR method based on four features.

The differences between the two methods mentioned above are consequence of an analysis based in general aspects of movements involved in the set of actions and the selected features to recognize the action. Then, although the number of features depends on the set of actions, it does not increase linearly with the number of actions because one or more actions can share features, that is, they can be distinguished using the same features or a subset of them. Therefore, effectiveness of the method depends mainly of the previous analysis performed on the set of actions to recognize. The following sections describe in detail each sub-process of the HAR method based in 6 features (Method 2).

4.3 Pre-processing

The aim of the preprocessing stage is to obtain binary snippets to feed the *BB tracking*. This stage consists of two main steps: a) Silhouette extraction and b) Generation of snippets sequences.

4.3.1 Silhouette Extraction

In this step, binary silhouettes are gotten using simple segmentation technics for each frame of all video streams belonging to each action dataset. For the three datasets which are the focus of attention of the research (Weizmann dataset, UIUC dataset and i3Dpost dataset) silhouettes are obtained by using the background subtraction technique and the Otsu thresholding method (both described in section 3.3). In the case of the evaluation datasets used in this work, the background does not have significant variations and then these simple technics can be used.

The pseudocode for silhouette extraction is provided in Listing Algorithm 1 and examples of silhouettes extracted for Weizmann, UIUC, and i3DPost datasets are shown in Figure 4.3. to Figure 4.5.

Algorithm 1: Silhouette extraction

input : Action video sequence
output: binary silhouette

Begin
Given a video of k frames
    $Bg \leftarrow Background\ image\ in\ grayscale$
    for *frame* = 1 to k
        $Fr \leftarrow frame\ image\ in\ grayscale$
        $Fr\_minus\_Bg \leftarrow (Fr - Bg)$
        $th \leftarrow Otsu\ threshold\ of\ (Fr\_minus\_Bg)$
        $binary\ silhouette \leftarrow threshold\ (Fr\_minus\_Bg\ with\ th)$
    end
    $output \leftarrow binary\ silhouette$
End

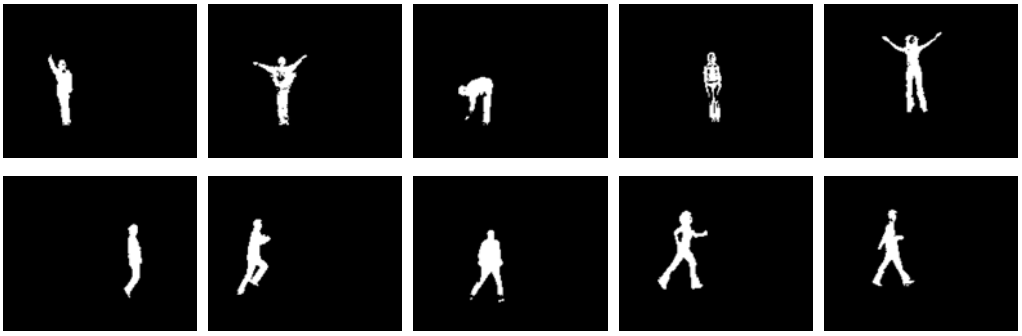


Figure 4.3. Examples of silhouettes extracted from the Weizmann dataset.

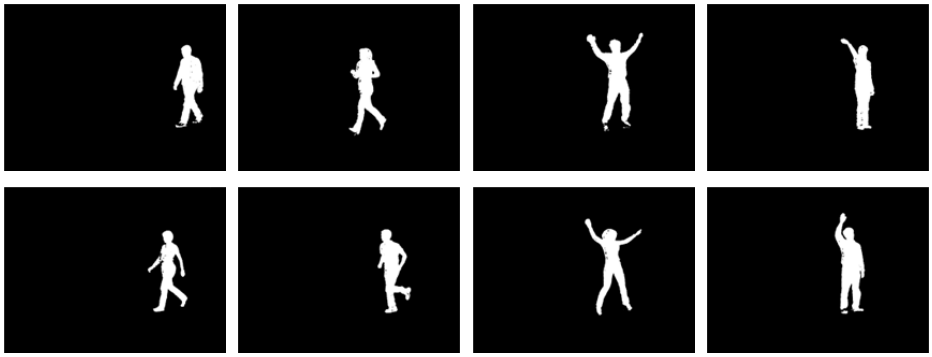


Figure 4.4. Examples of silhouettes extracted from UIUC dataset.

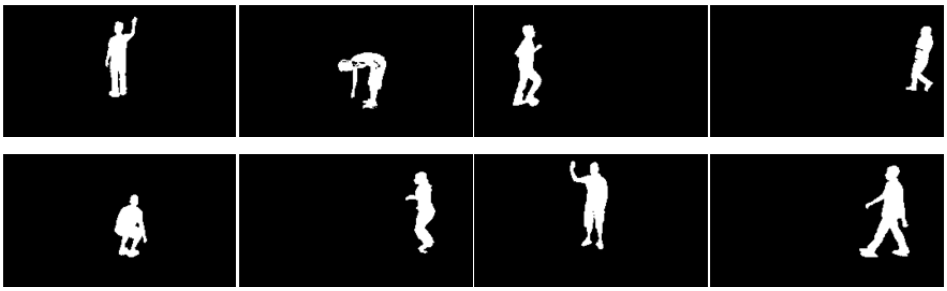


Figure 4.5. Examples of silhouettes obtained for i3DPost dataset.

4.3.2 Generation of Snippet Sequences

The sequences of binary masks resulting from "silhouette extraction" step are automatically divided into smaller sequences called *snippets*. Then, the proposed approach operates on sub-sequences instead of entire video sequences. Although several methods have used snippets [**59**,**27**,**29**,**36**], there is still not a current standard for the length of the snippet that should be used to reliably perform action recognition. While studies have revealed good results with very short snippets of lengths from 1 to 7 frames [**29**,**36**], in this work the snippet size is selected so that it includes at least one cycle of the action to be recognized because analyze few frames from the entire action cycle, specifically to recognize activities which are very similar (like "run" and "skip"), leaves out information essential for discrimination. Based on this argument, the snippet length is set to n, where n should include at least one cycle of the longest action cycle to be recognized. Therefore, the length of the longest cycle is measured for each dataset, resulting in a length of 30 frames for a Weizmann snippet and 40 frames for snippets belonging to UIUC and i3DPost datasets.

For the particular case of the i3Dpost dataset, that uses points of view, one additional pre-processing step is necessary. In this step a transformation of the video signal to maintain a fixed distance from the camera to the place where the action takes place is performed since the method proposed in this work is intended for scenarios where the actions are carried out in a plane parallel to the plane of the camera.

4.4 Bounding Box Tracking

The main objective of this sub-process is to track the object of interest (human silhouette) frame by frame in a simple and fast way. An estimate of the *Bounding Box search area*, symbolized as χ, is performed, thus decreasing the tracker's space of search. The estimation of χ is performed under the assumption that the human silhouette on each frame should be located in a neighboring region with regard to the region where it was located on a previous frame. Scanning of χ is done from the outside inwards, starting at all 4 edges enclosing the BB search area and ending at all 4 edges surrounding the human silhouette (BB edges).

To locate the top and bottom edges of the BB, scanning is done by row; and for the left and right edges of the BB, the sweep is done by column. An overview is showed in Figure 4.6 and the algorithm outline is resumed in Listing Algorithm 2.

The algorithm consists of three sequential steps, *first BB extraction*, *BB search area estimation* and *2nd to last BB extraction*. In frame 1, the tracker performs a search for the silhouette inside the *initial BB search area* χ_1 which is the whole initial frame and thus obtaining the *first Bounding Box* BB_1. In subsequent frames ($n \geq f > 1$), an estimated *BB search area* $\widetilde{\chi_f}$ is defined to reduce the tracker space of search. To get $\widetilde{\chi_f}$, the four edges of the BB on the previous frame BB_{f-1} are extended in ω pixels obtaining a new extended *BB search area*. Parameter ω is configurable and represents the maximum displacement in pixels of the BB between two consecutive frames.

Finally, at the *2nd to last BB extraction* stage, once the estimated *BB search area* $\widetilde{\chi_f}$ was obtained, the tracker scans within this area to get the BB on the current frame BB_f.

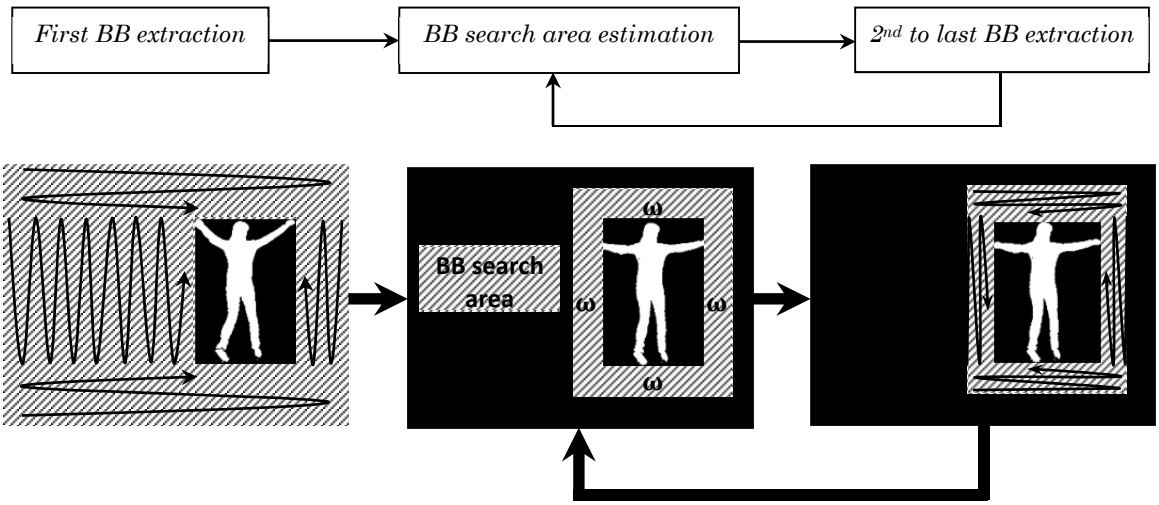


Figure 4.6. Outline of the BB tracker.

Algorithm 2: Bounding Box Tracking

Input : Snippet of n frames

Output: Bounding Box of each frame

1. **First BB extraction**

   At the outset (frame 1), the tracker scans the entire image (initial BB search area χ_1= size of frame) and gets the smallest BB (BB_1) enclosing the human silhouette.

2. **BB search area estimation**

   On the remaining frames (2 to n), the new BB search area estimated $\widetilde{\chi_f}$ is gotten by extending the dimensions of the BB on the previous frame BB_{f-1} in ω pixels around, $\widetilde{\chi_f} = BB_{f-1} + \omega$.

3. **2nd to last BB extraction**

   The tracker scans $\widetilde{\chi_f}$ and gets the actual BB (BB_f) enclosing the human silhouette.

The ω parameter is computed for the fastest analyzed action using equation (4.1).

$$\omega = \left\lfloor (elapsed\ time) * \begin{pmatrix} Avg\ speed\ of \\ fastest\ action \end{pmatrix} * \left(\frac{1}{image\ resolution} \right) \right\rfloor \quad (4.1)$$

Table 4.1 show the values of the parameters used in the computation of ω. The fastest action for three datasets is *"run"*, and the average value of *human running speed* used is $100m/18sec$[1]. In the case of the Weizmann dataset the *elapsed time* is $1/25\ sec$ and the $image\ resolution$ is $0.03\ meters\ per\ pixel$, and considering this data, the result is $\omega = 7\ pixels$. For i3DPost dataset $elapsed\ time = 1/25\ sec$, $image\ resolution = 0.004\ meters\ per\ pixel$ and the computed parameter $\omega = 60\ pixels$. Finally, for UIUC dataset, the value $\omega = 30\ pixels$ was obtained experimentally because the frame rate of dataset videos is not provided by the authors.

Table 4.1. Values of the parameters involved in computation of ω.

	ω *(pixels)*	*elapsed time (sec)*	*Avg speed of the fastest action* (m/sec)	*image resolution* $(m/pixel)$
Weizmann	7	1/25	5.5	0.03
UIUC [a]	30	–	–	–
i3DPost	60	1/25	5.5	0.004

[a] Parameter obtained experimentally.

4.5 Bounding Box Representation

One of the most important elements in the HAR system presented in this work is "the human body representation". This is because the proposed human model allowed us to select a feature set to represent action which is clear, reduced, and easy to compute. Thus, with this representation, the main objective is reduce the complex human form into a less detailed one that still retain enough information to

[1] Currently the Olympic record is fixed at 100m/9.63s then it considers an average person should be set to a half of that value.

distinguish the selected actions set. Therefore, based on the analysis of the human body movement involved in each of the actions to recognize, it observed that only three rectangular boxes are enough to model the human figure; *Bounding Box* (BB), *Knee Box* (KB) and *Feet Box* (FB). An example of this representation for action "jack" is shown in Figure 4.7. The first box (solid line) includes the full body (100% of the silhouette), the second box (dashed line) includes knees, legs and feet (30% of the lower part of the BB) and finally the third box (dotted line) includes feet (8% of the lowest part of the BB). The percentages for Knee Box and Feet Box were selected according to standard geometrical proportions of human body [**60**].

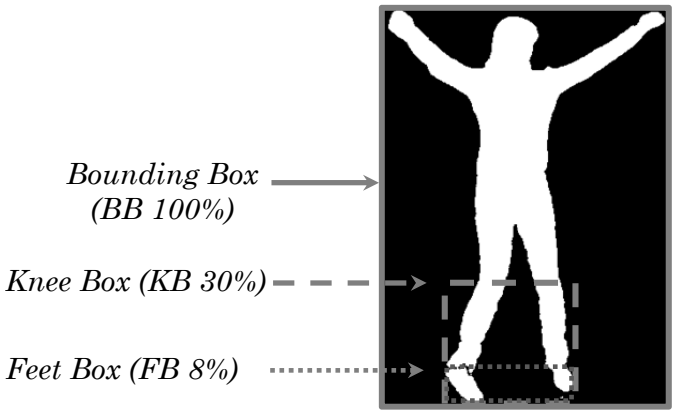


Figure 4.7. Model of human body based in three boxes: BB, KB and FB.

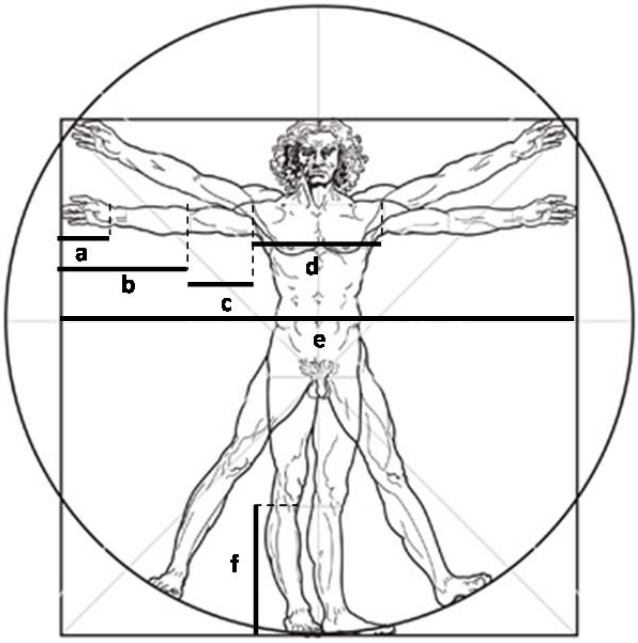
4.6 Features Extraction

Human action recognition in this work is performed using a set of features together with a hierarchical classifier system. Therefore, the features extraction stage is an important step in the proposed method. The features are based on domain knowledge and they are obtained following these three steps:

1. *Analysis*. General aspects of the movements involved in the process of each one of the human actions are observed and understood
2. *Selection*. What it need- or don't need to characterize each one of the actions is to identify both of their similarities and differences with other actions

3. *Formalization*. The information obtained from the two previous steps is used together with previous standard knowledge; for example, geometrical proportions and physical limitations of the human body to generate a set of formal features

Examples of geometrical proportions and physical limitations used in this work are shown in Figure 4.8.



a	The length of the hand is one-tenth of the height of a man
b	The length from elbow to the tip of the hand is equal to a quarter of the human height
c	the distance from the elbow to the armpit is one-eighth of the height of a man
d	The shoulders width is equal to a quarter of the human height
e	The length of the outspread arms is equal to human height
f	The length from the foot to the end of knee is equal to a quarter of the human height

Figure 4.8. Standard geometrical proportions of human body[2].

Some examples of domain knowledge are listed below

- The BB width growth in wave1 is lower than wave2 (Figure 4.9 rows 5 to 6)
- The BB width growth in wave2 and jack is similar, but the KB width is different (Figure 4.9 rows 4 and 5)

[2] Source :https://www.vectorstock.com/royalty-free-vector/vitruvian-man-vector-94736

- For the Run action both feet lose ground contact at some instant of time (Figure 4.9 row 3 columns 4 and 9)
- Walk action keeps at least one foot in ground contact at all times (Figure 4.9 row 2)
- For skip action both feet never touch the ground at the same time (Figure 4.9 row 1)

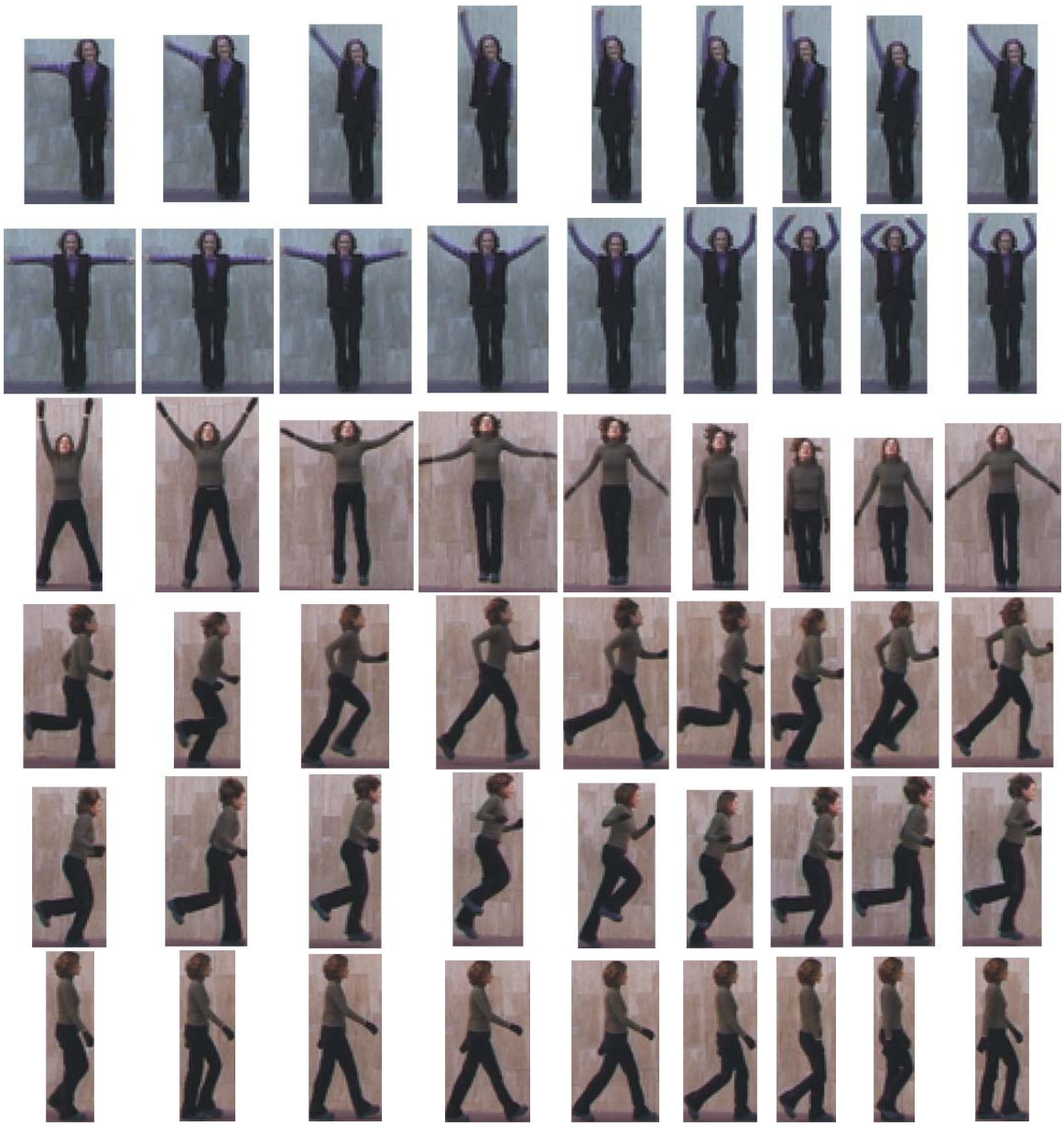


Figure 4.9. Examples of domain knowledge for person Lena of Weizmann dataset.

Based on the above analysis, it was determined that with a set of simple features extracted from three boxes and observing their evolution in time are enough to represent quantitatively the action. Then, two types of features were extracted: a) *Local features* that describe the morphology of the human body; and b) *Global features* that describe how movement unfolds.

4.6.1 Local Features

Six *local features* are extracted for each frame, three for BB (width, centroid abscissa coordinate and upper edge coordinate), two for KB (width and diagonal slope), and one for FB (number of feet planted on ground). The six *local features* are computed using rectangular coordinates. The four edge coordinates for *BB* are: left (x_{lft}), right (x_{rgt}), top (y_{top}) and bottom (y_{btm}); coordinates $k_{lft}, k_{rgt}, k_{top}, y_{btm}$ define *KB*; and $f_{lft}, f_{rgt}, f_{top}, y_{btm}$ define *FB*. The six local features are computed using equations (4.2) to (4.7).

Bounding Box

Width:

$$BB_{width} = \left| x_{rgt} - x_{lft} \right| \tag{4.2}$$

Centroid abscissa:

$$BB_{x_c} = x_{lft} + \left| \frac{x_{rgt} - x_{lft}}{2} \right| \tag{4.3}$$

Upper edge coordinate:

$$BB_{top} = y_{top} \tag{4.4}$$

Knee Box

Width:

$$KB_{width} = \left| \, k_{rgt} - k_{lft} \right| \tag{4.5}$$

Diagonal slope:

$$KB_{slope} = \left| \frac{y_{btm} - k_{top}}{k_{rgt} - k_{lft}} \right| \tag{4.6}$$

Feet Box

Number of feet on the ground:

$$feet \tag{4.7}$$

The pseudocode implementation of the local features extraction stage is provided in listing Algorithm 3,

Algorithm 3 : Local Features Extraction

input : Snippet of n Frames
output: Local Feature Vector L_F [$frame$ 1 to $n, feature$ 1 to 6]

```
1  Begin
2        for frame = 1 to n
3          if frame = 1
4              Find Bounding Box by scanning entire image
5          else
6              Get current Bounding Box from previous Bounding Box
7          end
8              find limits for Knee Box and Foot Box
9              compute local features using equations (4.2) to (4.7)
10             L_F [ frame ] ← Local Features 1 to 6
11       end
12 End
```

4.6.2 Global Features

This work uses 7 *global features* that are representative of one snippet. Four of them are extracted from BB (maximum width, range width, maximum horizontal displacement and maximum vertical scrolling), two from KB (maximum width and maximum diagonal slope) and one from FB (maximum number of feet touching the ground). The seven global features (G_F) are computed per snippet through an

analysis performed on all the n frames. Considering that i is the frame index and that one snippet contains n frames, global features are computed according to:

Bounding Box

Maximum width:

$$G_{F1} = \underset{i \in \{1,..,n\}}{max}(BB_{width}[i]) \tag{4.8}$$

Maximum horizontal shift:

$$G_{F2} = \underset{i \in \{1,..,n\}}{range}\left(BB_{x_c}[i]\right) \tag{4.9}$$

Range width:

$$G_{F3} = \underset{i \in \{1,..,n\}}{range}\left(BB_{width}[i]\right) \tag{4.10}$$

Maximum vertical shift:

$$G_{F4} = \underset{i \in \{1,..,n\}}{range}\left(BB_{top}[i]\right) \tag{4.11}$$

Knee Box

Maximum width of lower region:

$$G_{F5} = \underset{i \in \{1,..,n\}}{max}(KB_{width}[i]) \tag{4.12}$$

Maximum diagonal slope of lower region:

$$G_{F6} = \underset{i \in \{1,..,n\}}{max}\left(KB_{slope}[i]\right) \tag{4.13}$$

Feet Box

Maximum number of feet planted on the ground:

$$G_{F7} = \underset{i \in \{1,..,n\}}{max}(FB_{num}[i]) \tag{4.14}$$

Finally the computed global features are concatenated to get a single column feature vector per snippet

$$\boldsymbol{v} = [G_{F1},\ G_{F2},\ G_{F3},\ G_{F4},\ G_{F5},\ G_{F6},\ G_{F7}]^T$$

The pseudocode implementation of the Global Features extraction stage is provided in listing Algorithm 4.

Algorithm 4 : Global Feature Extraction

input : Local Feature Vector L_F [$frame\ 1\ to\ n, feature\ 1\ to\ 6$]
output: Global Feature Vector G_F [$feature\ 1\ to\ 7$]

```
Begin
    for feature index = 1 to 7
        compute Global Features using L_F vector and equations (4.8) to (4.14)
        G_F [ feature index ] ← Global Features
    end
End
```

4.7 Classifier

Features extraction is followed by a classification stage which captures the hierarchical nature of the set of human actions and model this set as a top-down hierarchy with classes inside classes (Figure 4.10). At each node there is a classifier with features as input data and action labels as outputs. There are three levels of classification with the top of the tree (root) placed at the first level. The number of levels in the path to classify a feature vector varies depending on leaf nodes located at the second and third levels of classification. Then, an action is classified after two or three levels of the hierarchy. One advantage of the model "classes inside classes" is that since few features are required by each classifier, these features can be plotted so that it is possible to visualize the fact that a linear hyperplane correctly separates classes. It was the reason for choosing classifiers that build linear hyperplanes (perceptron and SVMs).

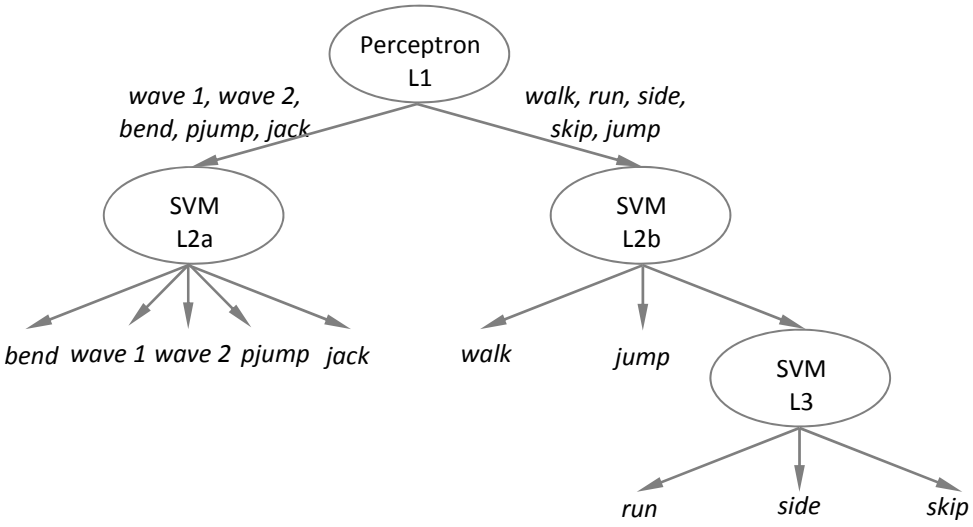


Figure 4.10. Hierarchical system of classifiers for the proposed method.

4.7.1 Architecture

The *hierarchical system of classifiers* consists of a structure of three levels of classification as it is shown in Figure 4.10 and Figure 4.11. At the highest level of the hierarchy there is one *perceptron* L1 (level 1) whose purposes are (1) to discriminate between the set of *actions carried out at the same place* $\alpha = \{wave1, wave2, bend, pjump, jack\}$ and the set of *actions where a displacement takes place* $\beta = \{walk, run, side, skip, jump\}$ and (2) to enable/disable the outputs of two *support vector machine* classifiers (SVM L2a or SVM L2b) at the second level of the hierarchical system of classifiers. At the second level of decision in the hierarchy, there are two SVMs, L2a and L2b. The aim of SVM L2a (Level 2a) is to separate set α into five unit sub-sets $\alpha_1 = \{wave1\}$, $\alpha_2 = \{wave2\}$, $\alpha_3 = \{bend\}$, $\alpha_4 = \{pjump\}$, and $\alpha_5 = \{jack\}$. The tasks of SVM L2b (Level 2b) are (1) to separate set β into three sub-sets $\beta_1 = \{walk\}$, $\beta_5 = \{jump\}$, and $\beta_{2,3,4} = \{run, side, skip\}$ and (2) to enable/disable the output of classifier SVM L3 at the third level of classification. Finally, at the lowest level of the hierarchy, the classifier SVM L3 (level 3) is found whose aim is to separate set $\beta_{2,3,4}$ into three unit sub-sets $\beta_2 = \{run\}$ $\beta_3 = \{side\}$ and $\beta_4 = \{skip\}$.

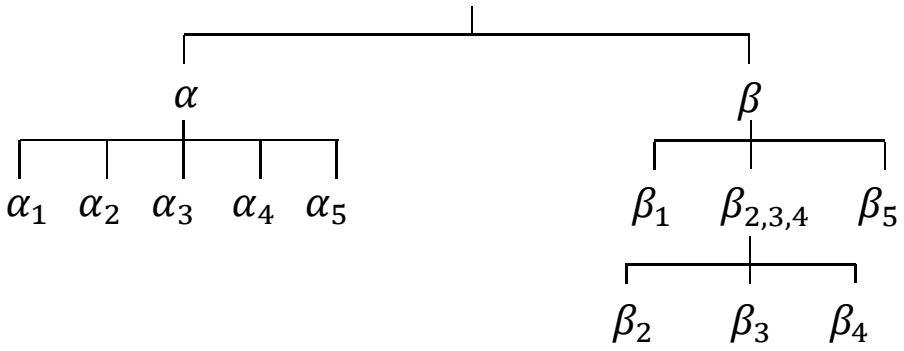


Figure 4.11. Top-down hierarchy of classification.

4.7.2 Training

The *hierarchical system of classifiers* consists of two types of classifiers: A binary classifier consisting of a single neuron (*Perceptron L1*) and three multi-class classifiers using linear kernels (*SVM L2a, SVM L2b and SVM L3*). The perceptron implementation used in this work is based in the Rosenblatt model [**57**], and the implementation of the three SVMs is based in the multiclass BSVM formulation described in [**58**]. Each classifier is independently training using a supervised learning model. The learning model uses a set of m training samples obtained from each dataset, where each sample is a pair $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^{m}$ consisting of a feature vector ($\boldsymbol{x}_i$), with 2 to 4 entries, which is a sub-set of the set of seven global features and a desired output value (action label d_i). Only two *global features* comprising the feature vector feed the *perceptron L1*, $[G_{F2}, G_{F3}]^T$; *SVM L2a* uses the vector of global features $[G_{F1}, G_{F3}, G_{F4}, G_{F5}]^T$; *SVM L2b* uses feature vector $[G_{F3}, G_{F4}, G_{F5}]^T$; and *SVM L3* uses two global features $[G_{F6}, G_{F7}]^T$.

4.7.3 Testing

In the testing stage, the hierarchical system of classifiers is used to assign an action label to an input snippet using partial outputs from each classifier. The hardware architecture of this system of classifiers is shown in Figure 4.12. Each multi-class SVM delivers its response through an output bus (vector). The SVM output bus feeds a multiplier which allows or prevents this bus to contribute in the

final action-labeling output vector. Also, a multiplier delivers its response through an output bus (vector) where each single bus line (vector entry) feeds a unit-step *activation function*. All activation function outputs are concatenated into an action-labeling output vector with 10 entries. There is one entry per action label and during classification one single label (entry) is activated. In Figure 4.12, it is shown that perceptron L1 generates output line y_0 which is used to enable/disable the output buses of classifiers SVMs L2b and L3 which are mutually exclusive. The inverted perceptron output $\overline{y_0}$ enables SVM L2a output bus y_1 which distinguishes among actions "wave1", "wave2", "bend", "pjump" and "jack". The output line y_0 enables SVM L2b output bus y_2 which is used to discriminate among actions "walk", "jump" and to activate/deactivate classifier SVM L3. SVM L3 output bus y_3 discriminates among actions "run", "skip", and "side". Outputs $y_1, y_2(entry\ 1)$, $y_2(entry\ 2)$ and y_3 are concatenated into a single 10-entry output vector (with an activation entry per action) to assign an unique action label to an input snippet.

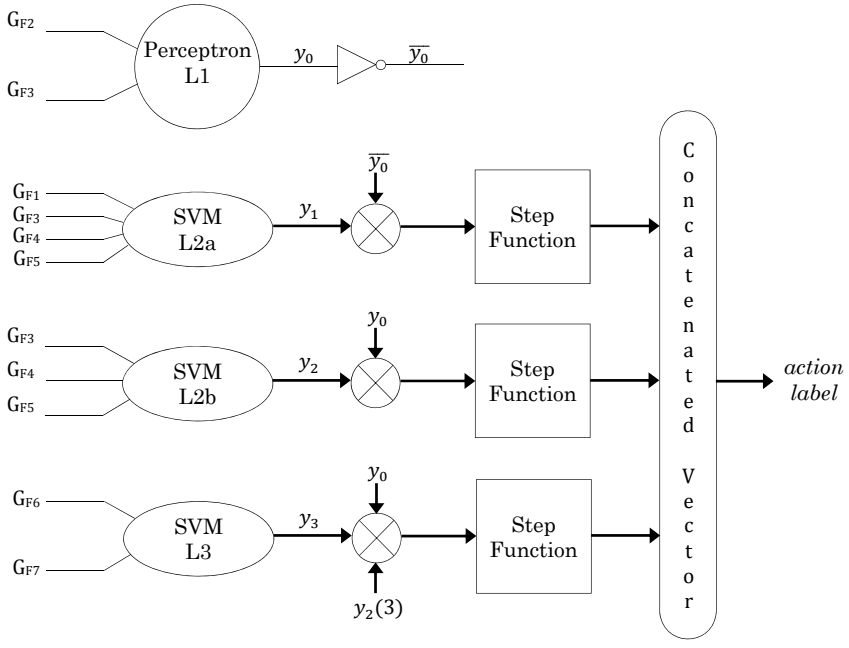


Figure 4.12. Procedure of action-labeling.

4.8 Summary

In this chapter, the proposed HAR method was presented. It consists of two phases, training and testing. In the first, feature vectors extracted from all video sequences used to train the method are used in the supervised learning process to configure a system of classifiers. In the testing stage, an unknown action video sequence is analyzed to obtain its feature vector and assigns an appropriate action label. Training and testing phases are very similar and then the most of the sub-processes used by both are the same, for example preprocessing, BB tracking, BB representation, features extraction and feature vector. In the *preprocessing* sub-process, binary silhouettes are obtained from full action videos using BS and Otsu thresholding method and then this silhouette sequences are divided in sub-sequences of *n* frames, known as *snippets*. Next, for each snippet frame, a *tracker* locates the smallest rectangle that encapsulates the human silhouette called *Bounding Box* (BB). Once the BB is defined, the human silhouette is represented using this BB rectangle, and two smaller rectangles KB and FB (Knee Box and Feet Box), contained within the original BB. Later, two types of features are extracted, *local features* and *global features*. *Local features* describe the morphology of a silhouette and are computed for each frame; and *global features* describe how movement unfolds and are computed for each snippet. Finally, global features are concatenated into a single *feature vector*. The last sub-process is the Classifier Model, training and testing phases differ in this because it is built in training phase and re-used in testing phase to predict the class of the unknown input video. The classifier model is a hierarchical system of classifiers which uses linear hyperplanes to separate the classes.

Chapter 5

5 Experiments and Results

The HAR method proposed in this work was evaluated on three publicly available datasets, Weizmann, UIUC and i3Dpost. It was implemented on two different development platforms, Matlab and Visual C++. In the first case, Matlab 13b of 64 bits was used and for second case Microsoft Visual C++ 2010 Express of 32 bits with the OpenCV library [**61**] was used.

This chapter is organized as follows. Section 5.1 provides a brief summary of the three dataset used in the evaluation. Experimental setup is described in section 5.2. Experiments and results of the Matlab implementation are presented in sections 5.3 and 5.4. The impact of pre-processing and BB tracking stages on method perform is measured in sections 5.5 and 5.6, respectively. In section 5.7, experiments and results of the C++ implementation are presented. The complexity analysis is given in section 5.8. Section 5.9 presents a multi-resolution evaluation and finally a summary concludes the chapter.

5.1 Datasets

Using publicly available datasets for evaluating the proposed method has two main objectives: 1) To facilitate comparison of the method with others and 2) give an

idea of the qualities of the same. Based on the review work provided by Chaquet *et al.* [**62**] , which makes a complete description of the most important datasets publicly available used in human action recognition, three datasets were selected to evaluate the proposed method. The purpose in selecting these datasets is to use common actions with significant distinct video resolution: *Weizmann dataset* [3] [**59**] at 180 x 144, *UIUC dataset* [4] [**63**] at 1024 x 768, and *i3Dpost* [5] [**64**] at 1920 x 1080.

5.1.1 Weizmann Dataset

This dataset consists of 90 *low resolution* video sequences (180 x 144, 25 frames per second) showing nine different persons with each person performing 10 different actions such as "run," "walk," "skip," "jumping-jack" (or "jack"), "jump-forward-on-two-legs" (or "jump"), "jump-in-place-on-two-legs" (or "pjump"), "gallopsideways" (or "side"), "wave-two-hands" (or "wave2"), "wave-one-hand" (or "wave1") and "bend". Some frames examples from different action sequences are shown in Figure 5.1.

Figure 5.1. Examples of frames extracted from Weizmann dataset.

5.1.2 University of Illinois at Urbana-Champaign (UIUC) Dataset

This dataset consists of 532 high resolution video sequences (1024 x 768) showing eight different persons with each person performing 14 different actions. However, experiments were conducted on the common subset of actions {"run,"

[3] Available at http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html , (Accessed: 8 March 2014).
[4] Available at http://vision.cs.uiuc.edu/projects/activity/ , (Accessed: 9 September 2014).
[5] Available at http://kahlan.eps.surrey.ac.uk/i3dpost_action/ ,(Accessed: 3 November 2014).

"walk," "jumping-jack" (or “jack”), "wave-one-hand" (or "wave1")}, which correspond to the actions for which the algorithm was designed. Some examples of frames from the UIUC database are shown in Figure 5.2.

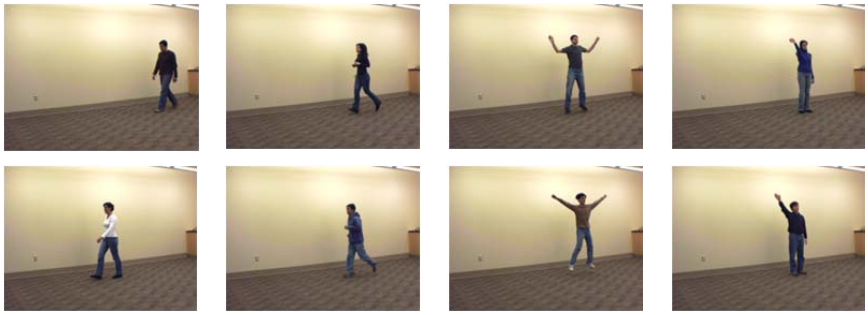


Figure 5.2. Examples of frames extracted from UIUC dataset.

5.1.3 i3DPost Multi-View Dataset

This dataset is a multi-view human action/interaction database and consists of 832 Full-HD resolution single-view video sequences (1920 x 1080, 25 frames per second) showing eight different persons (2 females and 6 males) with each person performing 12 different human motions (six actions and six interactions). The subjects have different sex, nationality, and significant differences in body sizes, and clothing. Experiments in this work were conducted on the subset of six actions {"run," "walk," "jump", "wave1", “bend” and “pjump”} corresponding to the ones in the Weizmann dataset, using two points of view which correspond to the actions for which the algorithm was designed. Some examples of video frames are shown in Figure 5.3.

Figure 5.3. Examples of frames extracted from i3DPost dataset.

5.2 Experimental Setup

Before describe in detail the different experiments carried out to measure the proposed method performance, experimental setup is presented. The method was evaluated in accuracy and speed using pre-segmented snippets of size 180x144x30 for Weizmann dataset, 1024x768x40 for UIUC dataset and 1920x1080x40 for i3Dpost dataset.

This work uses two standard protocols widely used in HAR literature; *Leave-One-Out Cross-Validation* (LOOCV) and 60%-40% to measure accuracy. Both protocols are particular cases for *leave-p-out-cross-validation* protocol and they were described in chapter 3.

Timing evaluation of the proposed method was measured on a notebook with Windows 7, an Intel iCore 7- 3610QM microprocessor at 2.3 Hz, 6GB RAM, a SATA-300 hard drive, and DTR 300Mbps using pre-segmented snippets carried out. The evaluation was divided in two parts. First, the average run time per snippet was measured, and based on this; the average run time per frame was computed.

Average run time per snippet was obtained by adding partial execution time for each stage of the proposed method. The average run time for a n-frame snippet was computed with equation (5.1)

$$Avg\ run\ time_{per\ snippet} = 1^{st} BB_{time} + BB\ 2^{nd} to\ last_{time} + n * LF_{time} + GF_{time} \quad (5.1)$$

Where $1^{st}\ BB_{time}$ is the average run time to extract the BB from the first snippet frame, $BB\ 2^{nd} to\ last_{time}$ is the average run time to obtain the BB for the remaining n – 1 snippet frames, LF_{time} is the average run time for *BB representation* and *local feature extraction*, and GF_{time} is the average run time for *global feature extraction*.

Average run time per frame was computed with equation (5.2), where $avg\ time_{per\ snippet}$ is the average run time per snippet obtained with equation (5.1) and n is the number of frames in the snippet.

$$avg\ time_{per\ frame} = \frac{avg\ time_{per\ snippet}}{n} \quad (5.2)$$

5.3 HAR Method Based in Four Features

In this section, the evaluation process for HAR method described in section 4.2 is presented. Four actions {*wave*1, *jack*, *walk and run*} and two different datasets, Weizmann and UIUC were used in the evaluation process.

Sub-sections 5.3.1 and 5.3.2 show the results of experiments used to measure the *Accuracy* and *Speed* of the method implemented in Matlab and sub-section 5.3.3 provides a comparison of these results with results obtained from other state-of-the-art methods.

5.3.1 Accuracy Evaluation

Method accuracy was assessed using two different technics, LOAO cross validation and 60%-40%. The classification results using both technics on Weizmann dataset were organized in the confusion matrices of size 4x4 showed in Figure 5.4. It can be seen that the CCR obtained for LOAO protocol is 100% and that for the case of Protocol (60% - 40%) the CCR obtained is 99.5%. It is observed that for the case of 60% - 40% protocol the highest confusion is located in action "jack". This is due to some snippets of jack action that are located at the class boundary.

	wave1	jack	walk	run
wave1	100.0	0.0	0.0	0.0
jack	0.0	100.0	0.0	0.0
walk	0.0	0.0	100.0	0.0
run	0.0	0.0	0.0	100.0

	wave1	jack	walk	run
wave1	100.0	0.0	0.0	0.0
jack	1.9	98.1	0.0	0.0
walk	0.0	0.0	100.0	0.0
run	0.0	0.0	0.1	99.9

Figure 5.4. Confusion matrix obtained for the Weizmann dataset using the LOAO Protocol (left) and Protocol 60% ─ 40% (right).

Confusion matrices, obtained for UIUC dataset using LOAO protocol and protocol 60% - 40%, are shown in Figure 5.5. The obtained CCR is 100% for the case of the LOAO Protocol and 99.35% when Protocol 60% - 40% was used. All actions

were correctly classified, except for the case of action *wave1* since this action is characterized by being executed in three different ways: 1) standard (hand over the head), 2) hand at shoulder level and outside the trunk, and 3) hand at shoulder level within the trunk; and the feature extraction stage is designed to just characterize the standard execution of action *wave1*; thus the confusion of the other two instances.

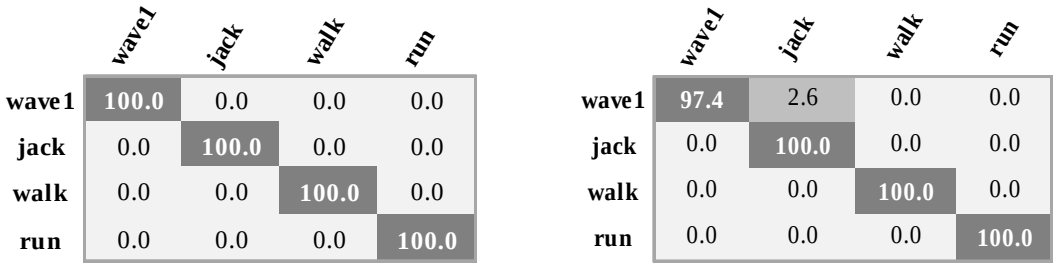


Figure 5.5. Confusion matrix obtained for the UIUC dataset using the LOAO Protocol (left) and Protocol 60% ─ 40% (right).

5.3.2 Speed Evaluation

The timing performance for the proposed method was obtained by measuring the average run time per snippet using equation (5.1). Results of the time contribution of each method stage to the average global run time, for both datasets, are shown in Table 5.1.

Table 5.1. Results of average run time evaluation per snippet.

Average run time	**Weizmann**		**UIUC**	
	ms	**%**	**ms**	**%**
$1^{st}BB_{time}$	5.50	13.24	15.8	20.74
$BB\ 2^{nd}to\ last_{time}$	29*0.13	9.07	39*1.18	60.42
LF_{time}	1.68	4.04	4.28	5.62
GF_{time}	30.65	73.65	10.07	13.22
Total	41.55	100	76.17	100

According to Table 5.1, it is observed that the average run time to get the first BB ($1^{st}BB_{time}$), for the two tested databases (Weizmann, UIUC) is more than 13 times the average run time to get the BB on any of the remaining frames ($BB\ 2^{nd}to\ last_{time}$) and the stage that contributes less to the overall processing time per snippet is the local features extraction (LF_{time}).

Some characteristics of HAR method applied to the Weizmann database are a processing rate of *722 fps* and processing run time of *1.385 ms* per frame with 180x144 = *25920 pixels* per frame. Characteristics for the case of the UIUC database are a processing rate of *526.31 fps*, and processing run time of *1.90 ms* per frame with 1024x768 = *786432 pixels* per frame. Then the number of pixels per frame in UIUC is more than *30 times* of a Weizmann frame while the average processing time is *1.37 times* that of a Weizmann frame. Thus, this clearly reflects the approach simplicity when the video resolution grows.

5.3.3 Comparison with other Methods

Table 5.2 and Table 5.3 present a comparison of the results of the proposed method versus those obtained from other state of the art methods. All methods were validated using the same protocol (LOOCV) and datasets (Weizmann and UIUC). According to Table 5.2, the CCR of the proposed method on Weizmann dataset is superior to that in [**36**], and comparable to methods reported in [**63**], [**32**], [**34**], and [**65**]. However, previous methods show some drawbacks if they are compared with the proposed method. First, the other methods requires between *27 and 170* times more features than the proposed method. Second, these methods use operations of higher complexity (powers, exponentials, trigonometric functions, square root among other) in comparison with those operations used by the proposed method (addition, subtraction, multiplication, divisions) and third, the processing rate obtained with the proposed method far exceeds that obtained by the other methods.

Table 5.2. Comparison of the proposed method with other methods for Weizmann dataset.

Methods	Year	Features	Accuracy (%)	FPS
Minhas *et al.* [**36**]	2012	400	99.9	5
Lin *et al.* [**65**]	2009	512	100	—
Wang *et al.* [**34**]	2013	680	100	—
Ikizler *et al.* [**32**]	2009	108	100	0.92
Du Tran *et al.* [**63**]	2008	216	100	—
Proposed method	**2015**	**4**	**100**	**722**

Results for the UIUC dataset reported in Table 5.3, show that the proposed method obtain better *accuracy* and *processing speed* with far fewer features (54 times less features) in comparison to the other method.

Table 5.3. Comparison of the proposed method with other methods for UIUC dataset.

Methods	Year	Features	Accuracy (%)	FPS
Du Tran *et al.* [**63**]	2008	216	98.70	—
Proposed method	**2015**	**4**	**100**	**526.31**

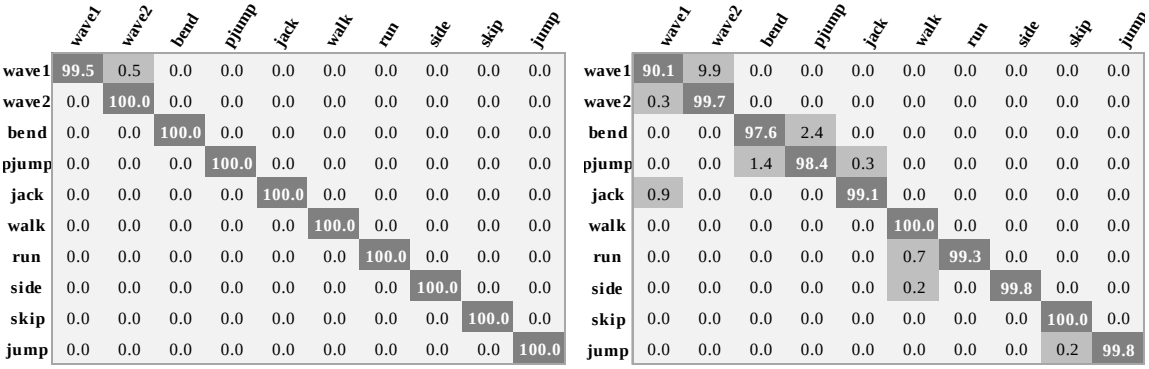
5.4 HAR Method Based in Six Features

In this section, the evaluation of HAR method described in section 4.1 is presented. Ten actions $\{wave1, wave2, bend, pjump, jack, walk, run, side, skip, jump\}$ and two different datasets, Weizmann and UIUC were used to evaluate the A*ccuracy* and *Speed* of the proposed method on Matlab platform.

5.4.1 Accuracy Evaluation

The proposed method was evaluated using two standard protocols, LOAO and 60%-40%. Confusion matrices, obtained for the Weizmann dataset using both protocols, are shown in Figure 5.6. It can be seen that the obtained accuracy for

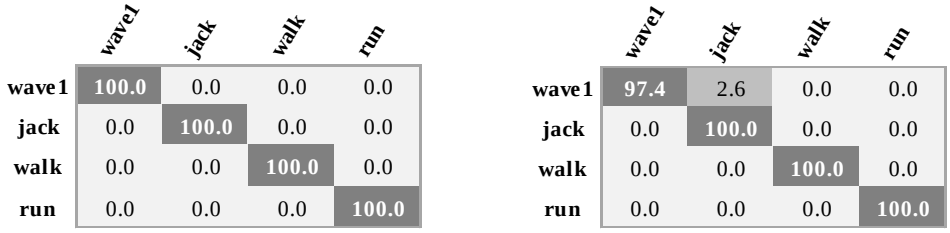
LOAO Protocol is 99.95% and 98.38% for the case of 60%-40%. It is observed that for both protocols the highest confusion is located in actions “wave1” and “wave2”.



	wave1	wave2	bend	pjump	jack	walk	run	side	skip	jump
wave1	99.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
wave2	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bend	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
pjump	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
jack	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
walk	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
run	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
side	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0
skip	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0
jump	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

	wave1	wave2	bend	pjump	jack	walk	run	side	skip	jump
wave1	90.1	9.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
wave2	0.3	99.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bend	0.0	0.0	97.6	2.4	0.0	0.0	0.0	0.0	0.0	0.0
pjump	0.0	0.0	1.4	98.4	0.3	0.0	0.0	0.0	0.0	0.0
jack	0.9	0.0	0.0	0.0	99.1	0.0	0.0	0.0	0.0	0.0
walk	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
run	0.0	0.0	0.0	0.0	0.0	0.7	99.3	0.0	0.0	0.0
side	0.0	0.0	0.0	0.0	0.0	0.2	0.0	99.8	0.0	0.0
skip	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0
jump	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	99.8

Figure 5.6. Confusion matrix obtained for the Weizmann dataset using the LOAO Protocol (left) and Protocol 60%-40% (right).



	wave1	jack	walk	run
wave1	100.0	0.0	0.0	0.0
jack	0.0	100.0	0.0	0.0
walk	0.0	0.0	100.0	0.0
run	0.0	0.0	0.0	100.0

	wave1	jack	walk	run
wave1	97.4	2.6	0.0	0.0
jack	0.0	100.0	0.0	0.0
walk	0.0	0.0	100.0	0.0
run	0.0	0.0	0.0	100.0

Figure 5.7. Confusion matrix obtained for the UIUC dataset using the LOAO Protocol (left) and Protocol 60%-40% (right).

Confusion matrices, obtained for UIUC dataset using LOAO Protocol and Protocol 60%-40%, are shown in Figure 5.7. The obtained accuracy is 100% for the case of the LOAO Protocol and 99.35% when Protocol 60%-40% is used. The subset of UIUC actions that is used for training and testing is {*wave1*, *jack*, *walk*, *run*} since this subset is common to both databases, Weizmann and UIUC. All actions were correctly classified, except for the case of action *wave1* since this action is characterized by being executed in three different ways: (1) standard (hand over the head), (2) hand at shoulder level and outside the trunk, and (3) hand at shoulder level within the trunk; and the feature extraction stage is designed to just characterize the standard execution of action *wave1*; thus the confusion of the other two instances.

5.4.2 Speed Evaluation

The average run time per snippet was obtained using equation (5.1). Results of the time contribution of each stage of the proposed HAR method to the average run time by snippet for Weizmann and UIUC dataset are shown in Table 5.4 and Figure 5.8.

Table 5.4. Results of time evaluation per snippet for Weizmann and UIUC datasets.

Average run time	**Weizmann (ms)**	**UIUC (ms)**
$1^{st} BB_{time}$	5.4	14.4
$BB\ 2^{nd} to\ last_{time}$	29*0.4	39*1.2
LF_{time}	30*4	40*4.5
GF_{time}	7.1	11.3
Total	144.1	252.5

According to Table 5.4, it is observed that the average run time to get the first BB is more than 12 times the average run time to get the BB on any of the remaining frames for both datasets.

Figure 5.8 shows *local features* (L_F) stage is the most contributes to the average run time per snippet of the proposed HAR method and *global features* (G_F) stage is the least contributes. Some characteristics of HAR applied to both datasets are a processing run time per frame of *4.8ms* for Weizmann and *6.31ms* for UIUC; a processing rate of *208.1 fps* with 144x180 = 25,920 pixels for Weizmann and *158.4 fps* for UIUC, with 1024x768 = 786,432 pixels per frame. Then, the number of pixels per frame for UIUC is *30 times* that of a Weizmann frame while the average processing time is *1.31 times* that of a Weizmann frame. Thus, this clearly reflects the simplicity of the proposed method when the video resolution grows.

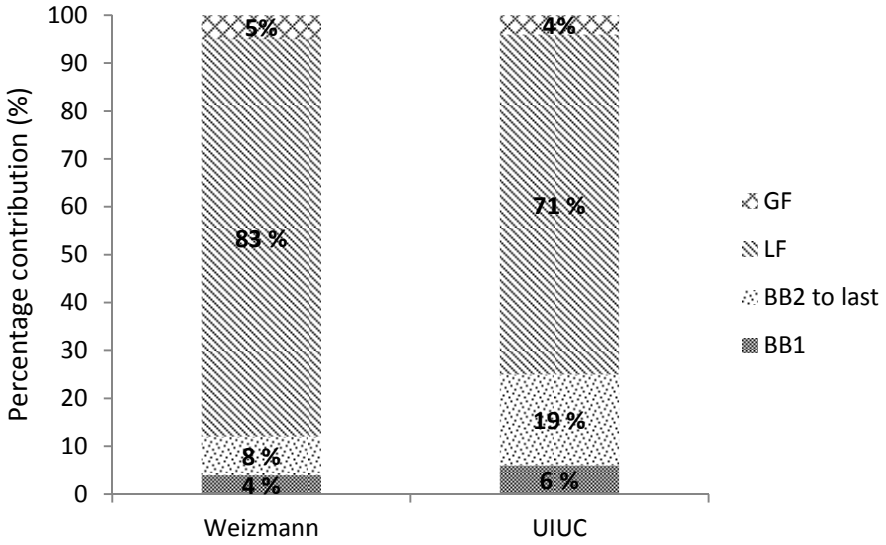


Figure 5.8. Percentage contributions of each stage to the average run time per snippet of the proposed HAR method for Weizmann and UIUC datasets.

5.4.3 Comparison with other Methods

Table 5.5 presents a comparison of the results of the proposed method versus those obtained from other state of the art methods evaluated on Weizmann dataset. All methods were validated using the same protocol. In terms of accuracy, the proposed method is superior to those presented in [**59**,**36**,**35**,**31**,**43**,**29**] and comparable to methods reported in [**32**,**34**,**65**]. However, for the case of the later methods, the proposed method is directly comparable only to methods reported in [**34**,**65**] because of the fact that the method reported in [**32**] discriminates only among nine of ten actions included in Weizmann dataset. The method in [**32**] does not include the "skip" action which is reported as the action that introduces the majority of errors in methods that use the complete set of 10 actions [**59**,**34**,**35**,**31**,**43**]. The methods, reported in [**34**,**65**], show some drawbacks if they are compared with the proposed method. First, the method requires a feature vector size between *85 and 113* times less than that from current comparable approaches in the state of the art literature. Second, these methods use operations of higher complexity (powers, exponentials, trigonometric functions, square root among other) in comparison with those operations used by the proposed method (addition,

subtraction, multiplication, divisions) and third, the processing rate obtained with the proposed method outperform that obtained by other methods.

Table 5.5. Comparison of the proposed method with other methods for Weizmann dataset.

Methods	Year	Features	Actions	Accuracy (%)	FPS
Bregonzio *et al.* [**35**]	2009	50	10	96.66	—
Gorelick *et al.* [**59**]	2007	$(m * n) * 7$ [6]	10	97.54	1.6
Marín *et al.* [**31**][a]	2012	1024	10	98.10	1.94
Guo *et al.* [**43**]	2009	$(m * n) * 13$	10	98.68	8.3
Schindler *et al.* [**29**]	2008	1000	9	99.60	—
Minhas *et al.* [**36**]	2012	400	9	99.9	5
Lin *et al.* [**65**]	2009	512	10	100	—
Wang *et al.* [**34**]	2013	680	10	100	—
Ikizler *et al.* [**32**]	2009	108	9	100	0.92
Proposed method	**2015**	**6**	**10**	**99.95**	**208.1**

[a] The computation time of BB is not included.

5.5 Importance of Pre-processing Stage

Some methods using silhouettes as input require high quality silhouettes images to achieve high performance because they use features based on the pixels that make up the silhouette or its edge. Then, some methods use additional techniques, such as morphological operations, filtering, or more sophisticated techniques in order to improve the silhouette quality. This chapter shows that in spite of poor quality of some silhouettes used to test the proposed method, achieves a high performance because the features are based on the bounding boxes corners and not on pixels inside silhouette.

[6] where m*n is the corresponding number of pixels in the human silhouette

The silhouettes, obtained through the Otsu method, are not always perfectly segmented from the background because of many factors; such as illumination variations, camera-person distance, similarities between clothing color and background color, and noise, thereby resulting in some low quality silhouettes. In order to give an idea of number and type of silhouettes used in the evaluation process all extracted silhouettes were analyzed and separated in two classes, *normal* and *abnormal*. Some examples of *abnormal* silhouettes for three datasets are displayed in Figure 5.9. Problems found in *abnormal* silhouettes are: (1) silhouettes mixed with shadows, (2) silhouettes with holes introduced by similarities between clothing color and background color, (3) silhouettes with disconnected body parts, (4) silhouettes with incomplete contours and (5) big-sized silhouettes scaled by decreasing the distance between camera and person. Percentage of occurrence obtained for *normal* and *abnormal* silhouettes for each dataset are showed in the bar graph of Figure 5.10, and percentage of occurrence for each type of *abnormal* silhouettes are showed in Table 5.6.

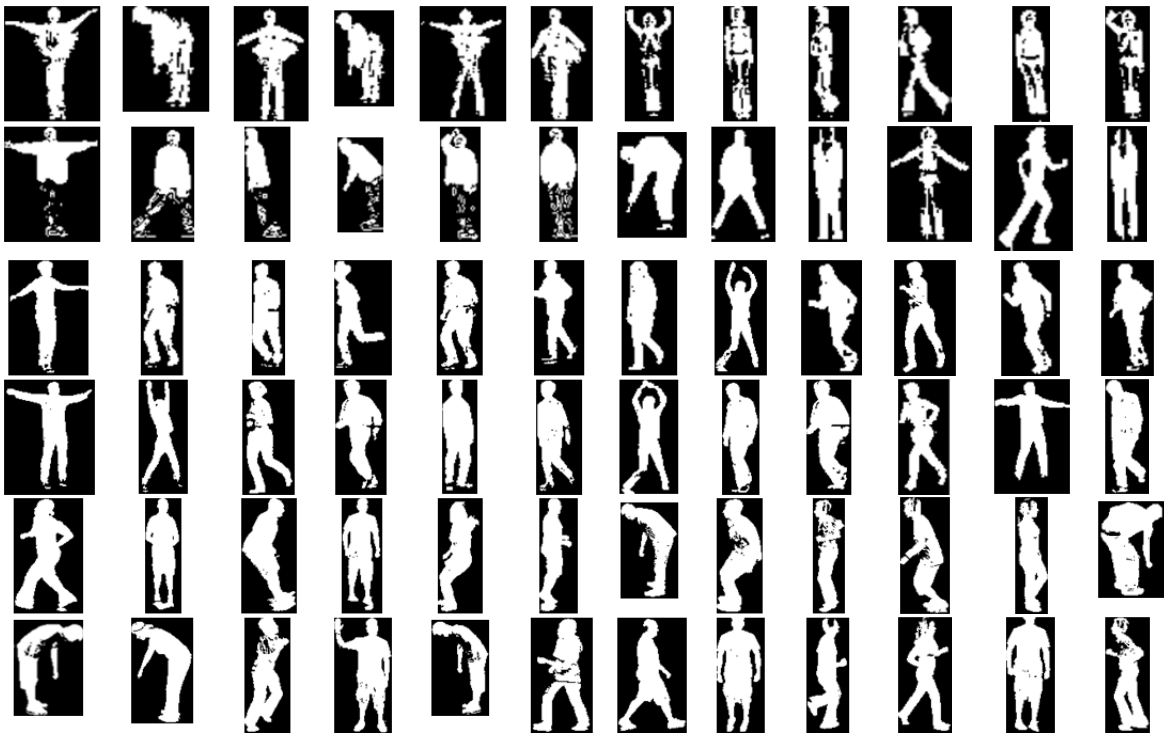


Figure 5.9. Examples of abnormal silhouettes extracted for three datasets, Weizmann (Rows 1-2), UIUC (rows 3-4) and i3DPost (rows 5-6).

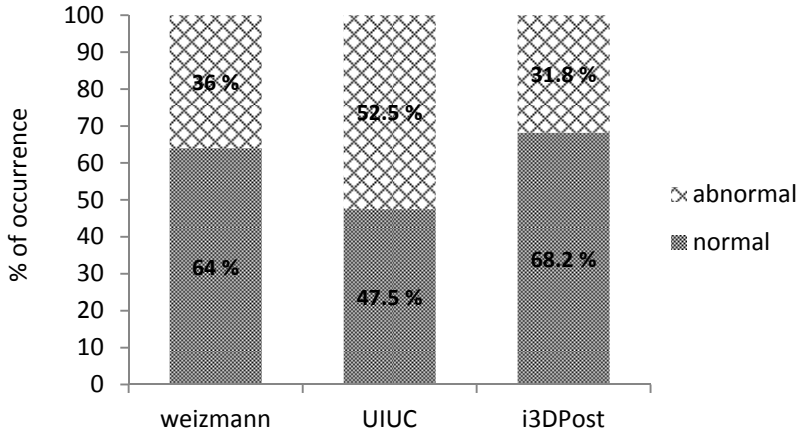


Figure 5.10. Percentages of normal and abnormal silhouettes.

Table 5.6. Percentage of occurrence for each type of abnormal silhouettes.

	Weizmann (%)	**UIUC** (%)	**I3DPost** (%)
Shadows	2	3.2	19.9
Holes	26	3.5	4.5
Disconnected parts	6	25.5	4.8
Incomplete contours	1	19.5	1.4
Big-sizes	1	0.8	1.2
Total	36	52.5	31.8

To evaluate how the quality of extracted silhouettes impact to the overall performance of the proposed method on Weizmann dataset, *normal* extracted silhouettes (64% of silhouettes) are used for training whereas abnormal extracted silhouettes (36%) are used just for testing. The obtained accuracy is 99.64% and the corresponding confusion matrix is shown in Figure 5.11. At snippet level the set of all normal frames becomes 40% of all snippets, and the set of snippets with abnormal silhouettes amounts to 60%, leading to the use of only 40% of all snippets for training and 60% for testing. The 3.6% of misclassified actions (as shown in the second row and first column of the confusion matrix) correspond to 7 snippets of action *wave2* executed by person *Shahar*; where action *wave2* is classified as action *wave1*. This misclassification takes place since the number of snippets for training (40%) is considerably smaller than the number of snippets for testing (60%) where

the last ones are characterized by features which are not used to learn the hyper-plane of the SVM classifier that separates the two action classes, *wave1* and *wave2*. Instead, the classifier is tested with these features which are located very close to the learnt hyper-plane as it can be observed in Figure 5.12.

	wave1	wave2	bend	pjump	jack	walk	run	side	skip	jump
wave1	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
wave2	3.6	96.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bend	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
pjump	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
jack	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
walk	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
run	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
side	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0
skip	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0
jump	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

Figure 5.11. Confusion matrix obtained for Protocol 40% - 60% with normal silhouettes (40% of all snippets) used for training and abnormal silhouettes (60% of all snippets) used for testing.

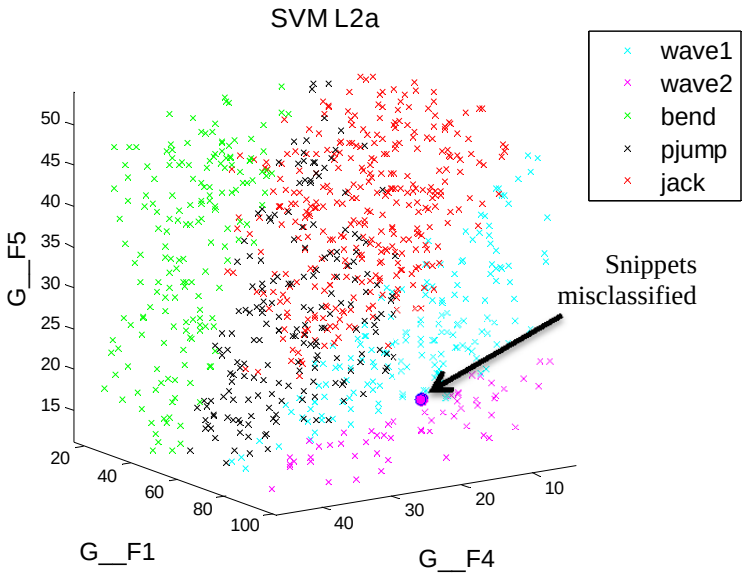


Figure 5.12. Decision maps with critical snippet feature vectors corresponding to misclassified actions.

In conclusion, in spite of the poor quality of some silhouettes, the proposed approach achieves a high performance because of the fact that features are obtained from the four corners of the BB enclosing the silhouette and not on pixels inside the silhouette. Moreover, features are computed as body proportions. Therefore, feature extraction is not affected by noisy silhouettes as long as they maintain their correct proportion between width and height.

5.6 Importance of Bounding Box Tracking

The core method, proposed in this work, is simple enough, and it relies on some stages, such as the BB tracking step. In this section, an analysis of how the tracking errors affect the performance of the proposed HAR method is carried out.

As it was explained in section 4.4, the proper operation of the tracking algorithm depends on parameter ω since this parameter is used to define the region where the tracker search and locates the BB (*BB search area*) for frames 2 to n. Then, incorrect values of this parameter introduce errors when parts of the human silhouette lie outside the *BB search area* such as in the cases of "hands out", as they are shown in the left and middle parts of Figure 5.13 and "feet out" as they are shown in the right part of Figure 5.13. These errors happen typically in actions which have fast displacement such as "jack" and "run".

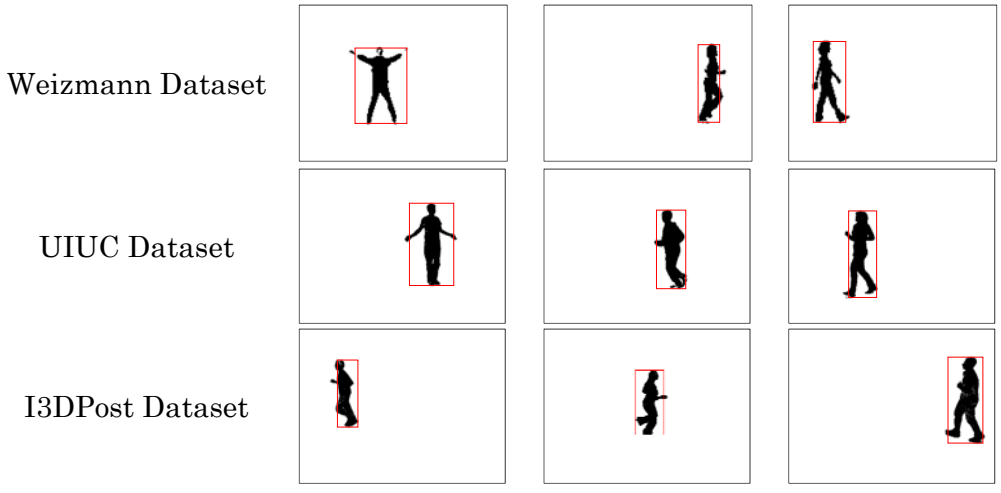


Figure 5.13. Examples of errors in BB tracking introduced by incorrect ω values such as "hands out" (left part and middle part) and "feet out" in (right part).

An analysis of how the parameter ω affects the method accuracy was performed on Matlab using gradual variations of $\pm\left(\frac{\omega_0}{3}\right)$ respect to ω_0, where ω_0 is the original value obtained in section 4.4 for each dataset, 7 pixels for Weizmann dataset, 30 pixels for UIUC dataset and 60 pixels for i3Dpost dataset. Results of accuracy evaluation using LOAO for the three datasets are showed in Table 5.7. According to this table, the accuracy for Weizmann dataset was kept unchanged for $\omega = 9$ pixels but the accuracy for $\omega = 5$ descended 0.64% respect to obtained for $\omega = 7$. The confusion matrix obtained is showed in left part of Figure 5.14. For the cases of the UIUC and i3Dpost datasets the results showed that the accuracy is maintained unchanged respect to ω_0 for $\omega = \omega_0 \pm \frac{\omega_0}{3}$ despite the fact that the BB tracking already presents errors as those shown in Figure 5.13.

Because of the fact that the reduction in accuracy was not perceived on two datasets, experiments decreasing the ω value were conducted on UIUC dataset until observe a change in the accuracy. This change was reached until $\omega = 17$ pixels which represents a decrease of 43.4% respect of ω_0 value. The method accuracy obtained was 98.75% and the confusion matrix is shown in the right side of Figure 5.14.

	wave1	wave2	bend	pjump	jack	walk	run	side	skip	jump
wave1	99.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
wave2	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bend	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
pjump	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
jack	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
walk	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
run	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
side	0.0	0.0	0.0	0.0	0.0	6.4	0.0	93.6	0.0	0.0
skip	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0
jump	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

Weizmann at w=5

	wave1	jack	walk	run
wave1	97.5	0.0	0.0	2.5
jack	0.0	100.0	0.0	0.0
walk	0.0	0.0	100.0	0.0
run	2.5	0.0	0.0	97.5

UIUC at w=17

Figure 5.14. Confusion matrix using the LOOCV protocol with errors introduced by the BB tracking.

Therefore, in order to perceive a decrease of only 1.2% in the method accuracy, it was necessary to decrease the ω value at 43.4%.

These results suggest that the proposed HAR method is robust to tracking errors because computation of the proposed feature set is based on human body proportions.

For timing analysis of how the parameter ω affects the performance of the proposed method, the average *BB search area* and the average run time for scan this area ($BB2^{nd}\ to\ last_{time}$) per frame were computed on Matlab using gradual variations of $\pm\left(\frac{\omega_0}{3}\right)$ respect to ω_0. Results for three datasets are shown in Table 5.7. According to this table, it observe that when ω_0 value increment or decrement 28.5%, *BB search area* changes less than a 34% and run time for one *BB 2nd to last* change a 5% for Weizmann dataset. When ω_0 value increment or decrement 33.3%, *BB search area* changes less than a 39%, and run time for one *BB 2nd to last* change less than 14% for UIUC. Finally, when ω_0 value increment or decrement 33.3%, *BB search area* changes less than a 38%, and run time for one *BB 2nd to last* change less than 18% for i3DPost. Thus, although the increase in value of ω causes a considerable increase in the *BB search area*, the run time for one *BB 2nd to last* increases only a fraction of the ω percentage and *BB search area* increase.

Table 5.7. Results of analysis for different ω values at the BB tracking for Weizmann, UIUC and i3DPost datasets.

	Weizmann			**UIUC**			**i3DPost**		
ω value (pixels)	*5*	*7*	*9*	*20*	*30*	*40*	*40*	*60*	*80*
Accuracy (%)	99.31	99.95	99.95	100	100	100	99	99	99
BB search area (pixels)	1012	1472	1965	21140	32910	45480	93432	144948	199664
$BB2\ to\ last_{time}$ (ms)	29*0.38	29*0.4	29*0.42	39*1.05	39*1.2	39*1.36	39*3.2	39*3.9	39*4.5

5.7 Multi-Dataset Evaluation on C++

This section uses three different datasets to assess the *Accuracy-Speed* performance of the proposed method, Weizmann, UIUC and i3DPost. The purpose in selecting these datasets is to use common actions, different body sizes, different

clothing, and significant distinct video resolution: Weizmann dataset 180x144, UIUC dataset 1024x768, and i3DPost dataset 1920x1080.

The method was implemented and evaluated on C++ platform. The evaluation process is organized in three sub-sections. Section 5.7.1 includes experiments and results of accuracy evaluation. Section 5.7.2 obtains algorithm timing evaluation on C++ platform and compares this with the evaluation obtained on Matlab platform, and finally section 5.7.3 provides a comparison between results obtained with the proposed method and other state-of-the-art methods.

5.7.1 Accuracy

In order to visualize and measure the method accuracy; experiments using LOAO protocol were carried out for each dataset. *Confusion matrices* obtained from these experiments are showed in Figure 5.15. These data show the accuracy based on the LOAO protocol is 99.95% for the whole Weizmann dataset, 100% for UIUC dataset, and 99% for i3DPost dataset.

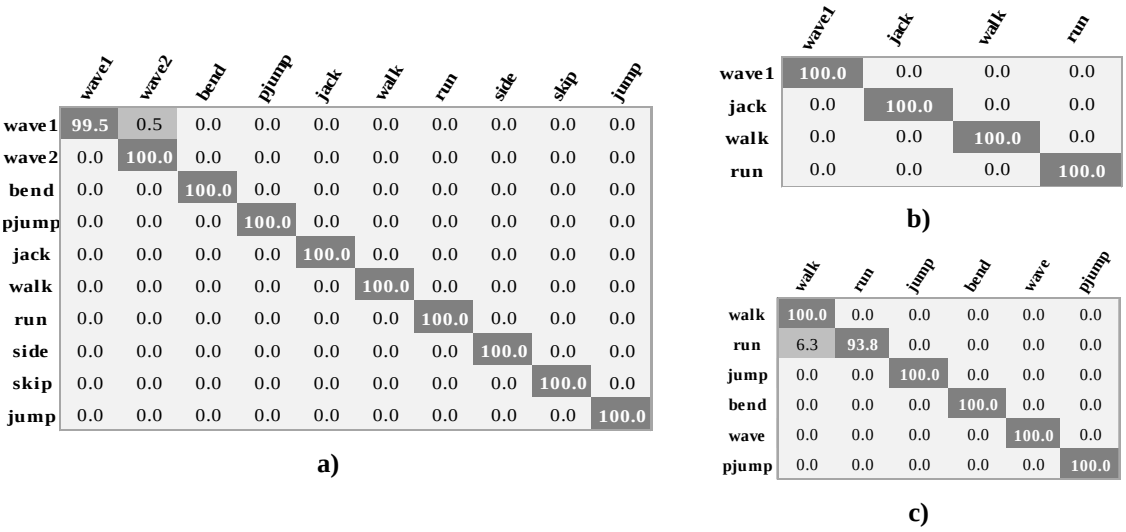


Figure 5.15. Confusion matrix obtained for a) Weizmann, b) UIUC and c) i3DPost datasets using LOAO.

Confusion Matrix obtained from Weizmann dataset show, the biggest confusion takes place in “wave 1” action. It happened because, some snippets of Denis actor include features at the limit of the class that defines “wave 1”, and the SVM wrongly

classifies them when these critical snippets are not part of the training set for the LOAO case. In the case of UIUC dataset, the maximum accuracy was obtained (CCR=100%). Finally, for the case of i3DPost, the worst confusion takes place at "run". This is due to the fact that some actors do "fast walk" instead of "running". Therefore, extracted features, from "fast walk" action, are confused with those of "run" action.

5.7.2 Speed

In this section, the algorithm time performance is computed for each dataset without applying additional optimizations in hardware or software. First, the average run time per snippet is measured and, then, based on this, the average run time per frame is obtained.

Average run time *per snippet* was computed with equation (5.1). Run time results for each algorithm stage per dataset are shown in Table 5.8 for both implementations (Matlab y C++). According to Table 5.8, which compares speed in a Matlab implementation with that in a C++ implementation, it observe a speedup from 7.9 to 82.1 times per snippet on C++. Regarding the ratio of average run time, to extract the first BB, compared with that required for subsequent BB extraction for Weizmann, UIUC, and i3DPost datasets is: 22, 12, and 6.8 times in Matlab; and 15, 30.7 and 9.8 times for C++ implementation. Therefore, estimation of a subsequent BB based on the current tracked BB significantly reduces processing time to track the BB from the second frame to the last one.

Table 5.8. Results of average run time per snippet for three datasets on Matlab and C++.

	Matlab			C++		
Average run time	**Weizmann (ms)**	**UIUC (ms)**	**i3DPost (ms)**	**Weizmann (ms)**	**UIUC (ms)**	**i3DPost (ms)**
$1^{st}\ BB_{time}$	4.4	14.4	28.1	0.15	2.15	3.84
$BB\ 2^{nd} to\ last_{time}$	29x0.2	39x1.2	39x4.1	29x0.01	39x0.07	39x0.39
LF_{time}	117	164	220	1.2	11.6	34.8
GF_{time}	7.5	18	19.2	0.0009	0.0009	0.0103
Total	134.74	243.19	427.15	1.64	16.48	53.86

Figure 5.16 shows percentage bar graphs for the average time per stage for the three datasets on Matlab and C++ implementations. For both implementations, the *local feature extraction* stage takes most of the time while *global feature extraction* is the stage that takes the least time. As frame size increases, time percentage contribution diminishes for *local feature extraction*. BB tracking time percentage contribution increases as frame size increases, but at a fraction of the expected 30.3 and 80 times, to 0.94 and 1.6 (3.1%, and 2%), respectively for UIUC, and i3DPost datasets for the C++ implementation.

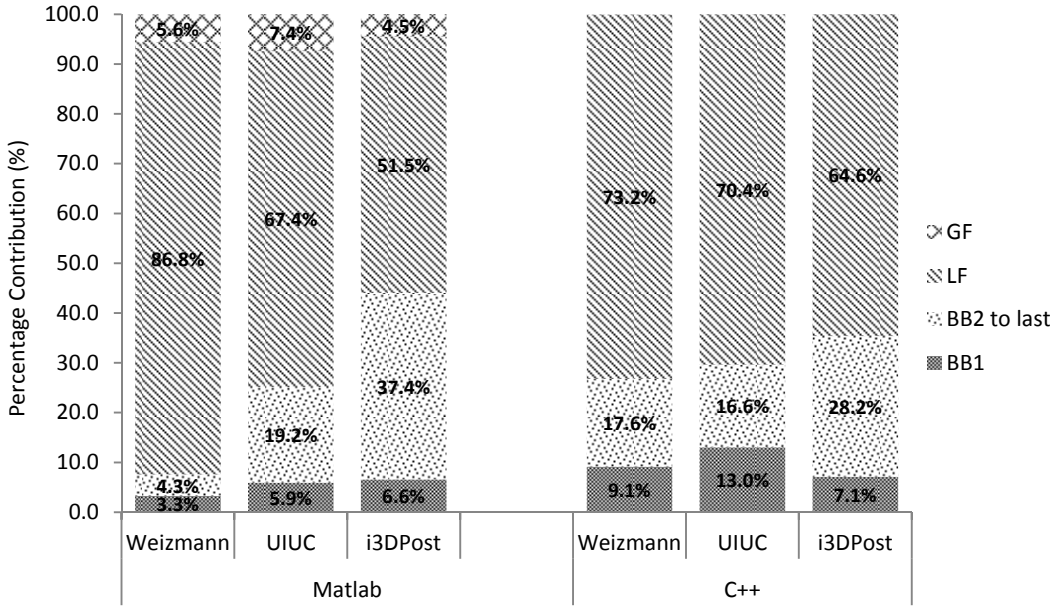


Figure 5.16. Percentage contributions by stage to the average run time per snippet.

Average run time *per frame* is computed using equation (5.2) for each dataset and implementation; results are shown in Table 5.9. It shows that while the number of pixels per frame in UIUC is higher than 30 times that in a Weizmann frame, and that the number of pixels per frame in i3DPost is higher than 80 times that of a Weizmann frame, the average run-time per frame in UIUC and C++ is 8.2 times that required to process a Weizmann frame and the average run-time per frame in i3DPost and C++ is 27 times that required to process a Weizmann frame. Therefore, these data clearly reflect the simplicity of the proposed method when video resolution is increased. Regarding, processing capacity of the proposed system exceeds from 29.7 to 731.3 times the frame rate at which the video was recorded.

Finally, the method achieves an average processing rate of more than *700 fps* at full-HD resolution images and over *18 282 fps* at low resolution images.

Table 5.9. Meaningful results of proposed HAR method.

	Matlab			**C++**		
	Weizmann	**UIUC**	**I3DPost**	**Weizmann**	**UIUC**	**I3DPost**
$No.\ of\ pixels_{per\ frame}(pixels)$	25 920	786 432	2 073 600	25 920	786 432	2 073 600
pixel grow	1	30.3	80.8	1	30.3	80.0
$Avg\ run\ time_{per\ frame}(ms)$	4.49	6.08	10.68	0.05	0.41	1.35
Avg run time grow	1	1.4	2.4	1	8.2	27.0
Processing rate (fps)	222	164	93	18 282	2 427	742

Figure 5.17 shows the percentages of contribution of both feature extraction stages (local and global) and BB tracking to the average run time per frame. Similar to the percentage of contribution of average run time per *snippet*, it is observed that the contribution of the extraction of features to the total decreases and the contribution of BB grows as the resolution is increased.

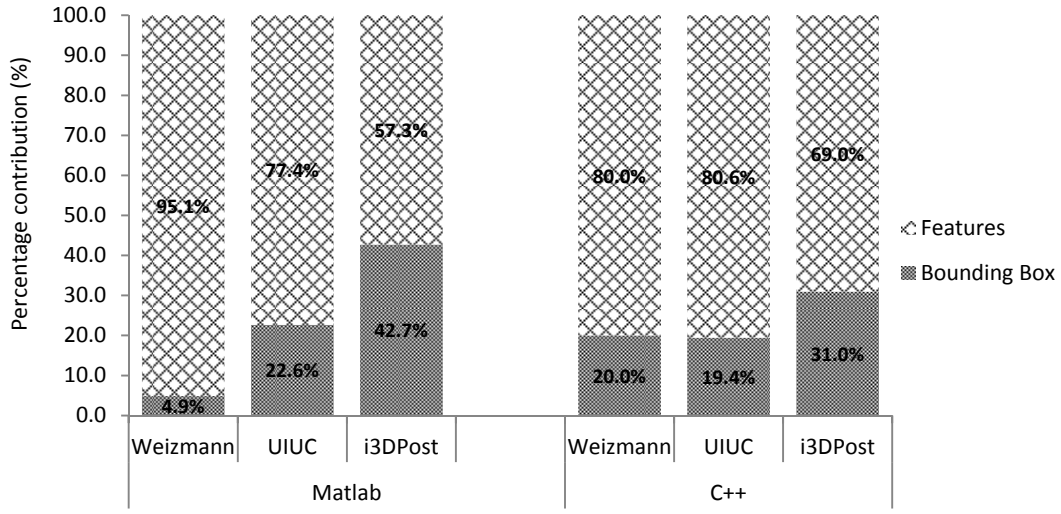


Figure 5.17. Percentage contributions by stage to average run time per frame.

5.7.3 Comparison

In this section, a comparison of the results obtained by the proposed method and those obtained from other state-of-the-art-methods in terms of *accuracy* (CCR) and *processing capacity* (fps) on the three datasets (Weizmann, UIUC and I3DPost) is presented. All methods were evaluated using the same datasets and protocol. Moreover, for a *fair comparison*, the proposed method like the compared methods that use silhouettes as input, use the binary silhouettes provided by the dataset or these are obtained previously by silhouette extraction techniques.

5.7.3.1 Weizmann Dataset

According to Table 5.10, it is observed that the results obtained by the proposed method are superior to those reported in [**40**,**41**,**39**,**48**,**38**,**50**,**49**] in *accuracy* (CCR) and *processing capacity* (fps). With regard to the reported methods in [**63**,**66**,**67**] the proposed method is the only one which quantifies the execution time, achieving a processing capacity of *18 282 fps* which exceed up *731 times* the frame rate at which the video was recorded and in terms of accuracy the proposed method is comparable to these methods. Then, the proposed method is superior to other methods in *Accuracy-Speed performance*. Moreover, the proposed method is better to [**63**,**66**,**67**] in different aspects. As an example, the method, reported in [**67**], uses a feature vector with *108 entries* while the method, reported in [**63**], is characterized by a feature vector with *216 entries*, and the method in [**66**] uses two types of features (low level and mid-level); where just the size for the mid-level feature vector is *1125*. These numbers exceed by more than *18 times* the size of the feature vector of the proposed method, which is *6*. Another aspect is that those methods, which discriminate among 9 out of 10 actions, included in the Weizmann dataset, do not include the "skip" action since it usually gets high recognition error rates and also reduces to a large extent the effectiveness of the recognition of other actions. According to Table 5.10, not all methods use the same traditional protocol Leave-One-Actor-Out Cross-Validation (LOAO), some use the amended version Leave-One-Sequence-Out Cross Validation (LOSO) which does not verify the robustness of

actor-variance and therefore gets better rates of classification. Finally, methods [**48**,**38**] compute the processing time only for one stage of method (tracking stage).

Table 5.10. Comparison of the proposed method with other methods for the Weizmann dataset.

Method	Year	Input	Actions	Protocol	CCR (%)	FPS
Ikizler & Duygulu [**67**]	2007	silhouettes	9	LOSO	100	—
Tran and Sorokin [**63**]	2008	silhouettes	10	LOSO	100	—
Fathi & Mori [**66**]	2008	images	10	LOSO	100	—
Cheema *et al.* [**39**]	2011	silhouettes	9	LOSO	91.60	56
Hernandez *et al.* [**38**] [a]	2011	images	10	LOAO	90.30	98
Sadek *et al* [**49**]	2012	images	10	LOAO	97.80	18
Chaaraoui *et al.* [**40**]	2013	silhouettes	9	LOSO	92.80	124
Chaaraoui & Flórez-Revuelta [**41**]	2014	silhouettes	10	LOAO	97.80	263
Hernández *et al.* [**48**][a]	2014	silhouettes	10	—	96.60	32.6
Rahman *et al.* [**50**][b]	2014	silhouettes	10	LOAO	95.56	11.62
Proposed method	**2015**	**silhouettes**	**10**	**LOAO**	**99.95**	**18 282**

[a] Processing capacity was measured only for the " Tracking Stage".

[b] Processing capacity was measured only for the" Feature Extraction Stage".

5.7.3.2 UIUC Dataset

Results on the UIUC, reported in Table 5.11, show that the proposed method exceeds other methods in *accuracy* and *processing capacity* (fps); however, the proposed method was evaluated on 4 out of 14 actions, included in the UIUC dataset, since only these actions are common to those of the Weizmann dataset for which the algorithm was originally designed. Furthermore methods in [**63**,**47**] downsize the ROI to 120x120 pixels which is the 30% of the average of the original ROI area.

Table 5.11. Comparison of the proposed method with other methods for the UIUC.

Method	Year	Input	Actions	Protocol	CCR (%)	FPS
Tran & Sorokin [**63**]	2008	Silhouettes downsized to 120x120	14	LOAO	98.7	0.005 [a]
Kalhor *et al.* [**47**]	2014	Silhouettes downsized to 120x120	14	LOAO	94.52	50
Hernández *et al.* [**48**]	2014	images	13	—	99.58	32.6
Proposed method	**2015**	**silhouettes**	**4**	**LOAO**	**100**	**2 427**

[a] This information was taken from reference [**47**]

5.7.3.3 i3DPost Dataset

According to Table 5.12, the proposed method show the next advantages compared to the other methods. First, the proposed method is the only one which quantifies the execution time over i3DPost dataset, managing to process a maximum of *742 fps*. Second, the proposed method gets the best accuracy. Third, it does not require downsize the ROI to reduce computational effort. Fourth, the proposed method requires up to *1024 times* less features. Fifth, the method use only simple arithmetic operations to compute the features per frame (addition, subtraction, multiplication, divisions). Therefore, the proposed method achieves the best *overall performance* in comparison with other methods, where *"overall performance"* means *Accuracy-Speed-Computational Effort* performance. Moreover, the proposed method shows *real-time* performance which is an indispensable attribute for a HAR system to be used successfully in a large part of potential application areas.

Table 5.12. Comparison of the proposed method with other methods for the i3DPost .

Method	Year	Input	Actions	Features	Protocol	CCR (%)	FPS
Gkalelis *et al.* [**64**] [a]	2009	Silhouettes downsized to W x H	5	W x H	LOSO	90%	—
Gkalelis [**68**]	2009	Silhouettes downsized to 32 x 64	5	2048	LOAO	90%	—
Holte *et al.* [**69**] [b]	2011	Raw Images	6	2304 x U	LOAO	89.58%	—
Iosifidis *et al.* [**70**]	2012	Silhouettes downsized to 64 x 64	6	4096	LOAO	95.33%	—
Iosifidis *et al.* [**71**]	2013	Silhouettes downsized to 32 x 32	6	1024	LOAO	98.16%	—
Proposed method	**2015**	**Raw Silhouettes**	**6**	**6**	**LOAO**	**99%**	**742**

[a] Where W is the width and H is the height of the smallest BB.
[b] Where U is the number of bins in radial direction.

5.8 Complexity Analysis

The fact that technology in video cameras and displays are continually improving in terms of cost, size and sensors, allow us to foresee a constant growing in image quality and frame rate for the following years. Then, the complexity analysis is done with respect to frame size growth.

The complexity of the proposed HAR algorithm with respect to frame size is given in function of the *BBsearch_area* which was presented in section 4.4 and described in detail in this section.

For a given snippet, the *BBsearch_area* is the average total area (in pixels) to explore to locate the BB enclosing the human silhouette. This area is defined in two different conditions.

Frame 1: (no assumptions on where the silhouette is)

$$BBsearch\_area_{frame1} = frame\_size - BB\_size \tag{5.3}$$

Frame 2 to n: (the previous location of the silhouette is known)

$$BBsearch\_area_{frame\ 2\ to\ n} = xBB\_size - BB\_size \tag{5.4}$$

Where according to Figure 5.18, *frame_size* is the number of pixels per frame (W*H), *BB_size* is the number of pixels in a Bounding Box (BBw*BBh) and *xBB_size* is the number of pixels of the extended bounding box (xBBw*xBBh). Then for a given snippet of n frames the algorithm complexity with respect to *BBsearch_area* is

$$O\big[\big((W * H) - (BBw * BBh)\big) + (n-1) * \big(2\omega * (xBBw + BBh)\big)\big] \tag{5.5}$$

Or

$$O[(frame_{size} - BB_{size}) + (n-1) * (xBB_{size} - BB_{size})] = \tag{5.6}$$

$$O(BBsearch_{area\,frame1} + (n-1) * BBsearch\_area_{frame\ 2\ to\ n})$$

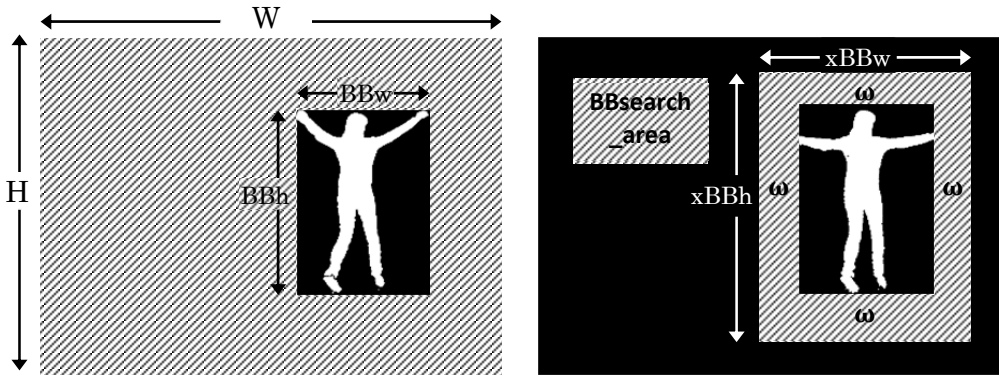
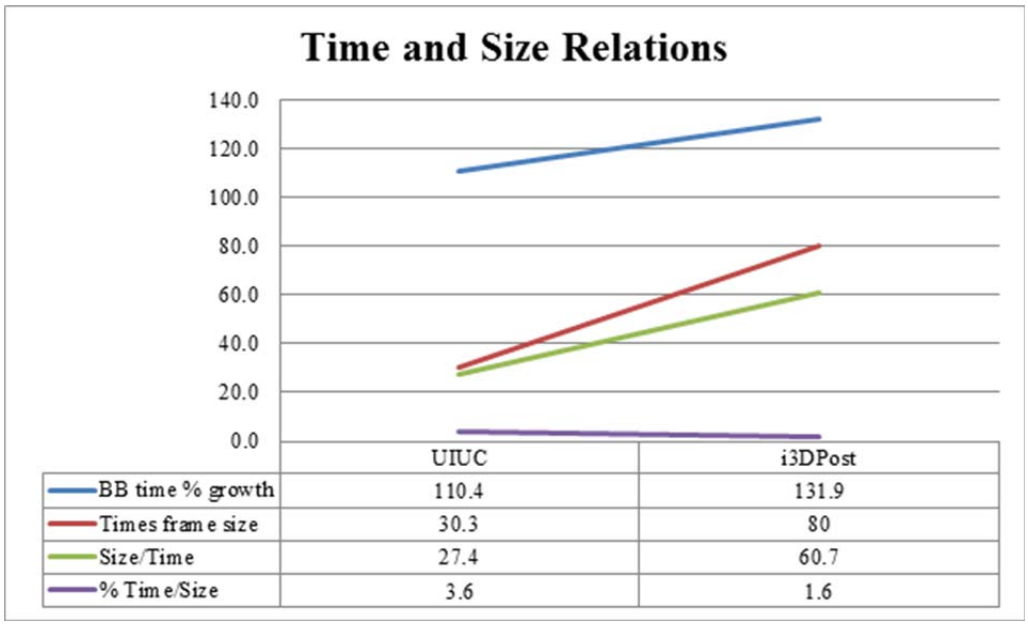


Figure 5.18. Bounding Box search area.

For the case of the frames 2 to *n*, the average total area (in pixels) to explore for the three datasets is lower than 6.9% of the total *frame_size*.

Because, the proposed algorithm works on *BBsearch_area* in what follows complexity with respect to *frame_size* is derived and how it relates to *BBsearch_area*.

The growth of *BBsearch_area* for UIUC and i3DPost datasets with respect to the Weizmann dataset are 22.4 times and 98.6 times, respectively. Considering the C++ implementation, the growth in execution time for bounding box extraction is a fraction of the growth in pixels for *BBsearch_area*, giving 8.3 times for UIUC, and 32.5 times for i3DPost that of the execution time for Weizmann. Both cases are about one third of their corresponding pixel growth. On average the actual execution is a fraction of the *BBsearch_area* pixels. Figure 5.19 puts all together with respect to *frame_size*, and all bounding box operations. The actual pixel percentage used for BB average computation per snippet is in the range from 1.6% to 3.6% or in the range from 27.4 to 60.7 times less pixels than that of the *frame_size* for UIUC and i3DPost datasets, respectively. In terms of the BigO notation this corresponds to $O(frame\_size^q)$ where $0 < q < 1$, which is known as fractional power complexity and it means the algorithm complexity is *sublinear*.



	UIUC	i3DPost
BB time % growth	110.4	131.9
Times frame size	30.3	80
Size/Time	27.4	60.7
% Time/Size	3.6	1.6

Figure 5.19. Bounding Box frame, size, and time relations.

5.9 Multi-Resolution Timing Evaluation

This section uses the i3DPost dataset at different video resolutions to assess the time performance of the method described in section 4.2. The processing time was measured on C++ implementation with the goal of analyze the behavior of the proposed method when the frame size in test videos is modified.

The original frame size of i3DPost dataset (1920x1080) was rescaled from 1/8 to 4 times (Figure 5.20) obtaining video resolutions from 32kpix to 33Mpix. The bicubic resizing algorithm was used and the aspect ratio was kept.

Once the i3DPost dataset has been resized, the average run time per snippet was computed for different video resolutions using equation (5.1). Then, equation (5.2) was used to obtain the average run time per frame, and finally the processing rate computed.

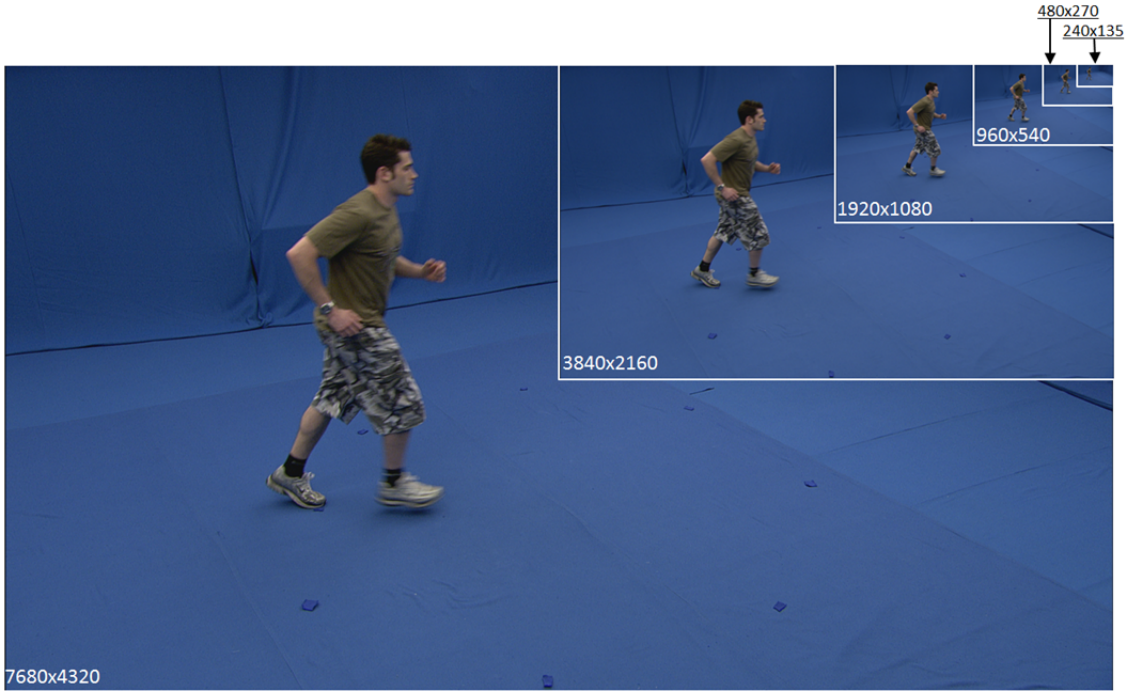


Figure 5.20. Resizing of i3DPost dataset.

Temporal evaluation results for different i3DPost dataset resolutions are shown in Table 5.13. Results show that despite the algorithm was evaluated in Full HD, 4K UHD and 8K UHD video resolution; the algorithm still shows real-time performance with a processing rate from *126,613 fps* in low resolution (240x135) videos to *46fps* in 8K UHD video resolution.

Table 5.13. Time performance for different i3dpost dataset resolutions.

	240x135	**480x270**	**960x540**	**1920x1080**	**3840x2160**	**7680x4320**
Avg run time $(ms)_{per\ frame}$	0.008	0.013	0.081	0.319	4.76	21.32
Processing rate (fps)	126,613	76,661	12,399	3,138	210	46

Bar graph in Figure 5.21 shows the contribution percentages for each stage to average run time per frame. It is observed that as frame size increases, percentage contribution of local features extraction run time decreases and the contribution of tracking grows. Finally, it must be emphasized that at HD video resolution, contribution of each stage becomes steady.

Largest contributor stage to total average run time per frame is tracking; which is according to complexity analysis and it contributes over 98% on all different resolutions.

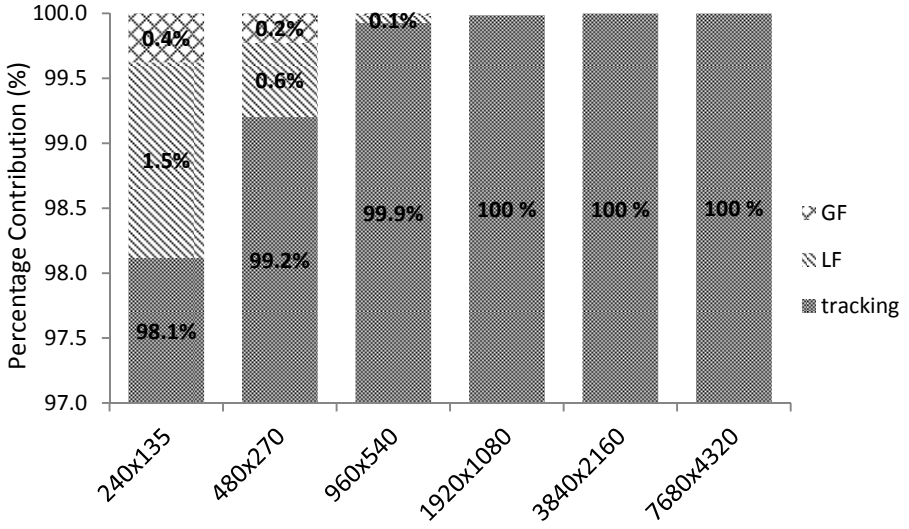


Figure 5.21. Percentage contributions of each stage to average run time for different i3DPost dataset resolutions.

5.10 Summary

This chapter presented the set of experiments carried out to measure the effectiveness of the proposed HAR method. The method was evaluated in accuracy and speed on three available publicly datasets, Weizmann, UIUC and i3Dpost. For accuracy evaluation, the LOOCV protocol was used and the confusion matrixes obtained. For the speed evaluation, the average run time of each stage of the proposed method was measured on Matlab and C++ implementations using a standard laptop.

Both versions of the proposed method were evaluated, the method based in four features and the method based in six features. Additionally, tolerance of proposed method to segmentation and tracking errors was assessed and finally, the algorithm complexity and the method's performance were evaluated when the video resolution is increased up to 8K UHD.

General results on the three datasets show the proposed HAR method is superior to the other state-of-the-art methods in processing capacity (18 282 fps, 2 427 fps and 742 fps) and comparable or superior in accuracy (99.95%, 100%, and 99%). It obtains high performance despite segmentation or tracking errors and to the best of our knowledge, the proposed method is the first one that shows *real-time* performance on videos up *8K UHD,* so it can easily deal with the constant improvement of the *image quality and resolution*.

Chapter 6

6 Conclusions and Future Work

6.1 Conclusions

Most of the methods, reported in HAR literature, are based on complex models which require the calculation of a very large number of parameters and high processing time. This work has shown that by incorporating knowledge of the natural domain of the problem in the HAR process, efficient recognition in *real-time* with significantly reduced information is achieved which is essential for large part of HAR application areas.

The presented method in this dissertation combines very simple technics together with clear and simple concepts to achieve high performance in *accuracy* and s*peed*. It uses BB tracking based estimation; a human body model based on three rectangular boxes (BB, KB, and FB), a reduced number of features (six per frame) which are clear, simple and easy to compute, and a hierarchical system of classifiers based on linear hyperplanes.

Three publicly available datasets with different resolution, Weizmann (180x144), UIUC (1024x768) and i3DPost (1920x1080) were used to evaluate the proposed method. Experimental results on these datasets show that the presented method is superior to other state of the art methods, in *processing capacity* (18 282 fps, 2 427

fps and 742 fps, respectively) and comparable or superior in *Accuracy* (99.95%, 100%, and 99%) with far fewer features (up to 1024 times less features). Additionally, the presented method has the following qualities that distinguish it from the other methods. It exceeds several times the frame rate at which the videos were recorded (up to 731 times in low resolution and 29 times in Full-HD). Only a fragment of the whole image is processed to track and locate the BB (less than 7% of the whole image). The ROI size does not have to be downsize to reduce computational load as it happens in other cases where the ROI area was downsized up to the 0.4% of average original ROI area. The method works on sub-sequences of the entire video (snippets) and it does not require high quality silhouettes to achieve high accuracy performance. The features are based on the BB parameters, and not on the intensity or amount of the pixels inside the ROI, the human silhouette or the silhouette contour. Then, the number of features does not depend on the resolution of the image. The method uses simple arithmetic operations (sums, subtractions, multiplications, and divisions) for feature extraction. The relationship *"processing time per frame"* versus *"video resolution"* is of fractional power complexity, and the proposed method is the first one that shows *real-time* performance on videos up *8K UHD,* so it can easily deal with the constant improvement of the *image quality and resolution.*

Finally, based on the attributes mentioned above, one can conclude that the method presented in this research work, promises a good future and can be easily implemented as an embedded system on a video camera making the video signal analysis possible in *real-time* and paving the way for the so-called *"edge processing"*.

6.2 Future Work

As it was mentioned in previous chapters, the recognition of human actions based in video is a quite extensive research topic, and although there have been great advances in the field still exists a long way to go before the proposed methods can operate efficiently in real life environments. Particularly in this dissertation there are many future research opportunities. However, only some of the most important points will be mentioned.

- Improve the segmentation technique employed for more complex environments
- Enlarge the set of actions to be recognized and extend the proposed method to analysis of videos involving activities
- Increase the number of points of view from which the action is observed (multi-view HAR)
- Explore the field of action recognition involving two or more people in the scene (multi-person HAR)

Appendix A

A. Video Surveillance Systems in Mexico City

With the aim of reducing the crime rate, increase the efficiency of the police, and speed the response time of emergency personnel, in October 25th of 2011 the Mexico City government inaugurated the "Centro de Control, Comando, Comunicación y Cómputo, Inteligencia, Investigación, Información e Integración" (C4i4) [**72**]. But while there are not enough eyes to constantly monitor the 8000 (as of 2014) video surveillance cameras installed in the city, they provide only an illusion of security.

The true data of this promise of security are:

The C4i4 (Figure A.1) concentrate the images from 8,000 video surveillance cameras installed in the City of Mexico, which are observed through 180 screens, and two screens are monitored per security person. Therefore; around 44.4 cameras are observed on each screen and an average of 88.8 cameras are monitored by a single person. Based on these data one can conclude the following:

- Surveillance cameras monitoring is intermittent and sporadic (it is not continuous)
- The number of observation screens and staff are insufficient
- The efficiency of monitoring staff is low because of the overwhelming data
- The actual chance that an emergency is observed on line by personnel is very slim; thus the alert time to emergency teams is not in time to prevent the event

Therefore; there is still a need for tools that can help improve the effectiveness of video surveillance systems.

Figure A.1. C4i4 Mexico City [7]

[7] Source: http://elfederalista.mx

Appendix B

B. Market Growth of HD Surveillance Cameras

The HD-surveillance camera was introduced for first time in 2009 and since its sales have been increased. The plot in Figure B.1 shows the expected growing of surveillance camera for next years [**73**].

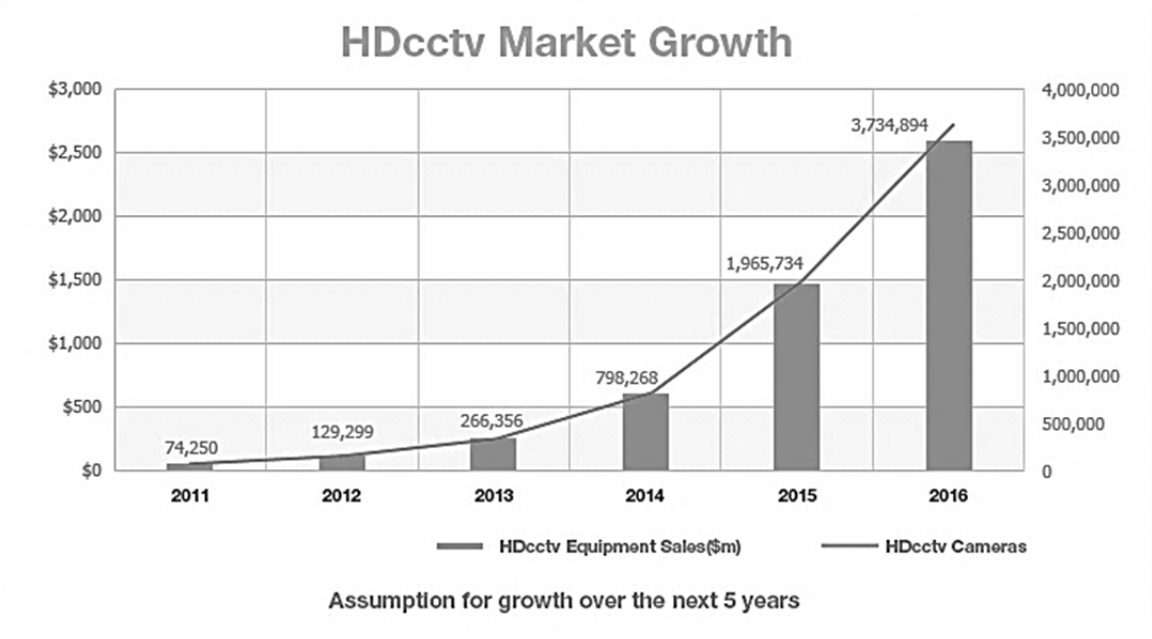


Figure B.1. Expected worldwide growing of surveillance camera sales[8]

[8] Source: http://www.webgate-usa.com

Appendix C

C. Data Growth of Video Surveillance

The video data is growing exponentially, and surveillance video has become the largest source. It is a consequence of the increase in the number of cameras installed around the world, the new high definition (HD) video cameras market, and a huge data of unimportant footage for example empty street, a dog crossing the street etc. According to

Figure C.1, the daily data in 2013 achieved 413 PB which is equivalent to fill 92.1 million of single sided, single layer DVDs or, it is four times the amount of photo and video data stored on Facebook as of February 2012, and its expected the surveillance data achieve 859PB in 2013. Then strategies are needed to reduce and optimize the storage and transmission of this huge amount of data [**74**].

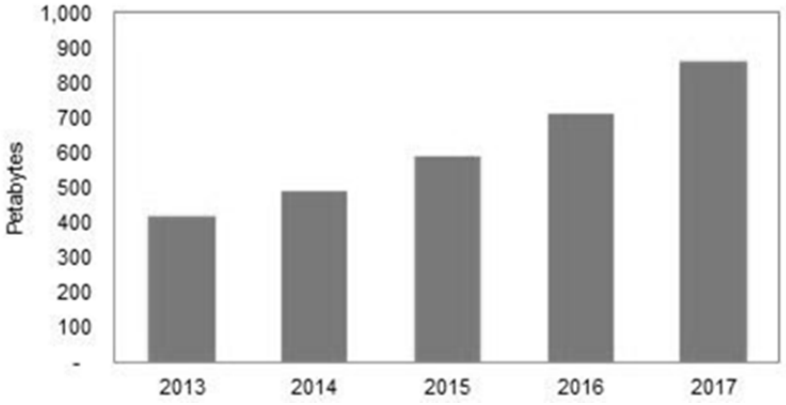


Figure C.1. Global data generated daily by surveillance cameras[9]

[9] Source: https://technology.ihs.com

Bibliography

[1] R. a. G. T. Klette, "Understanding human motion: A historic review," in *Human Motion*, R. K. D. M. Bodo Rosenhahn, Ed. Springer Netherlands, 2008, ch. 1, pp. 1-22.

[2] S. Herbert, L. McKernan, and individual contributors. Who's who of Victorian cinema. [Online]. http://www.victorian-cinema.net/news

[3] L. Mitchell, "The Man Who Stopped Time," *Stanford Magazine*, May 2001.

[4] S. Mamber, "Marey, the analytic, and the digital," in *Allegories of communication: Intermedial concerns from cinema to the digital*, J. Fullerton and J. Olsson, Eds. Rome, Italy: John Libbey Pub., 2006, pp. 83-91.

[5] W. Braune and F. O., *The Human gait*. Germany: Springer-Verlag Berlin Heidelberg, 1987.

[6] G. Johansson, "Visual Perception of Biological Motion and a Model for its," *Perception Psychophysics*, vol. 14, no. 2, pp. 201-211, 1973.

[7] Ó. D. Lara and M. A. Labrador, "A survey on Human Activity Recognition using Wearable Sensors," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 3, pp. 1192-1209, 2013.

[8] A. Ghali, A. S. Cunningham, and T. P. Pridmore, "Object and Event Recognition for Stroke Rehabilitation," in *In Proceedings of Visual Communications and Image*

Processing, Lugano, Switzerland, 2003, pp. 980-989.

[9] V. Di Salvo, A. Collins, B. McNeill, and M. Cardinale, "Validation of Prozone: A new video-based performance analysis system," *International Journal of Performance Analysis in Sport*, vol. 6, no. 1, pp. 108-119, 2006.

[10] G. Demiris, D. Parker Oliver, J. Giger, M. Skubic, and M. Rantz, "Oder adults' privacy considerations for visionbased recognition methods of eldercare applications," *Technology and Health Care 17*, vol. 17, no. 1, pp. 41-48, 2009.

[11] A. A. Chaaraoui, P. Climent Pérez, and F. Flórez Revuelta, "A review on vision techniques applied to human behaviour analysis for ambient-assisted living," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10873-10888, 2012.

[12] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce, "Automatic annotation of human actions in video," in *In Computer Vision, 2009 IEEE 12th International Conference on*, Kyoto, 2009, pp. 1491-1498.

[13] T. B. Moeslund, "Interacting with a virtual world through motion capture," in *Interaction in Virtual Inhabited 3D Worlds*. Berlin/New York: Springer-Verlag, 2000, ch. 11.

[14] C. S. Regazzoni, A. Cavallaro, Y. Wu, J. Konrad, and A. Hampapur, "Video analytics for surveillance: Theory and practice ," *Signal Processing Magazine, IEEE* , vol. 27, no. 5, pp. 16-17, Sep. 2010.

[15] V. Saligrama, J. Konrad, and P.-M. Jodoin, "Video anomaly identification," *Signal Processing Magazine, IEEE*, vol. 27, no. 5, pp. 18-33, 2010.

[16] D. Hambling. (2010, May) The Future of Surveillance? When Automated Brains Keep Watch 24/7. [Online]. http://www.popularmechanics.com/technology/security/how-to/a5806/future-of-surveillance-cameras/ . (Accessed: 18 march 2015).

[17] A. Hampapur, "Smart video surveillance for proactive security," *Signal Processing Magazine, IEEE*, vol. 25, no. 4, p. 136, 2008.

[18] T. Huang. "Surveillance Video: The Biggest Big Data", Computing Now, vol. 7, no. 2, Feb. 2014,IEEE Computer Society. [Online]. http://www.computer.org/web/computingnow/archive/february2014. (Accessed: 20 March 2015).

[19] J. Green. (2014, Dec.) Processing video at the edge with insights from behavioural analysis. [Online]. http://www.techworld.com/blog/machination/processing-video-at-edge-with-insights-from-behavioural-analysis-3590327/ . (Accessed: 18 March 2015).

[20] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2, pp. 90-126, 2006.

[21] R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing*, vol. 28, no. 6, pp. 976-990, 2010.

[22] X. Ji and H. Liu, "Advances in view-invariant human motion analysis: a review," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 40, no. 1, pp. 13-24, 2010.

[23] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 16, 2011.

[24] S. R. Ke, et al., "A Review on Video-Based Human Activity Recognition," *Computers*, vol. 2, no. 2, pp. 88-131, 2013.

[25] P. Afsar, P. Cortez, and H. Santos, "Automatic visual detection of human behaviour: A review from 2000 to 2014," *Expert Systems with Applications*, vol. 42, no. 20, pp. 6935-6956, Nov. 2015.

[26] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. ,. Basri, "Actions as space-time shapes," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on* , Beijing, 2005.

[27] Kai Guo, Prakash Ishwar, and Janusz Konrad, "Action Recognition in Video by Covariance Matching of Silhouette Tunnels," in *Computer Graphics and Image*

Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on, Rio de Janiero, 2009.

[28] D. Y. Chen, "Efficient polygonal posture representation and action recognition," *Electronics Letters*, vol. 47, no. 2, pp. 101-103, 2011.

[29] K. Schindler and L. Van Gool, "Action Snippets: How many frames does human action recognition require?," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* , Anchorage, AK , 2008.

[30] K. G. Derpanis, M. Sizintsev, K. J. Cannons, and R. P. Wildes, "Action Spotting and Recognition Based on a Spatiotemporal Orientation Analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* , vol. 35, no. 3, pp. 527-540, 2013.

[31] M. J. Marín Jiménez, N. P. de la Blanca, and M. Á. Mendoza, "Human action recognition from simple feature pooling," *Pattern Analysis and Applications Springer-Verlag*, pp. 1-20, 2012.

[32] N. Ikizler and P. Duygulu, "Histogram of oriented rectangles: A new pose descriptor for human action recognition," *Image and Vision Computing*, vol. 27, no. 10, p. 1515–1526, 2009.

[33] S. Baysal and P. Duygulu, "A line based pose representation for human action recognition," *Signal Processing: Image Communication*, vol. 28, no. 5, pp. 458-471, 2013.

[34] J. Wang, P. Liu, M. F. H. She, A. Kouzani, and S. Nahavandi, "Supervised learning probabilistic Latent Semantic Analysis for human motion analysis," *Neurocomputing*, vol. 100, no. 16, pp. 134-143, 2013.

[35] M. Bregonzio, S. Gong, and T. Xiang, "Recognising action as clouds of space-time interest points," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* , Miami, FL, 2009.

[36] R. Minhas, A. A. Mohammed, and Q. M. J. Wu, "Incremental Learning in Human Action Recognition Based on Snippets," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol. 22, no. 11, pp. 1529-1541, 2012.

[37] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 3, pp. 257-267, 2001.

[38] J. Hernández, A. Montemayor, J. Pantrigo, and A. Sánchez, "Human action recognition based on tracking features," in *Foundations on Natural and Artificial Computation*, La Palma, Canary Islands, Spain, 2011, pp. 471-480.

[39] S. Cheema, A. Eweiwi, C. Thurau, and C. Bauckhage, "Action recognition by learning discriminative key poses," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International*, 2011, pp. 1302-1309.

[40] A. A. Chaaraoui, P. Climent-Pérez, and F. Flórez-Revuelta, "Silhouette-based human action recognition using sequences of key poses," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 1799-1807, 2013.

[41] A. A. Chaaraoui and F. Flórez-Revuelta, "A Low-Dimensional Radial Silhouette-Based Feature for Fast Human Action Recognition Fusing Multiple Views," *International Scholarly Research Notices*, vol. 2014, 2014.

[42] P. Natarajan and R. Nevatia, "Online, Real-time Tracking and Recognition of Human Actions," in *Motion and video Computing, 2008. WMVC 2008. IEEE Workshop on*, 2008, pp. 1-8.

[43] P. Guo, "Real time human action recognition in a long video sequence," in *Proceedings of the 2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 248-255.

[44] H. Meng, N. Pears, and C. Bailey, "Recognizing human actions based on motion information and SVM," in *Intelligent Environments, 2006. IE 06. 2nd IET International Conference on*, 2006, pp. 239-245.

[45] B. Chakraborty, A. D. Bagdanov, and J. Gonzalez, "Towards Real-Time Human Action Recognition," in *Pattern Recognition and Imge Analysis*. Springer Berlin Heidelberg, 2009, pp. 425-432.

[46] S. Sadek, A. Al-Hamadi, B. Michaelis, and U. Sayed, "An Efficient Method for Real-Time Activity Recognition," in *Soft Computing and Pattern Recognition (SoCPaR), IEEE International Conference on*, 2010, pp. 69-74.

[47] D. Kalhor, I. Aris, I. A. Halin, and T. Moaini, "A Fast Approach for Human Action Recognition," in *Intelligent Systems, Modelling and Simulation (ISMS), IEEE International Conference on*, 2014.

[48] J. Hernández, R. Cabido, A. S. Montemayor, and J. J. Pantrigo, "Human activity recognition based on kinematic features," *Expert Systems*, vol. 31, no. 4, pp. 345-353, 2014.

[49] S. Sadek, A. Al-Hamadi, B. Michaelis, and U. Sayed, "A Fast Statistical Approach for Human Activity Recognition," *International Journal of Intelligence Science*, vol. 2, no. 1, pp. 9-15, 2012.

[50] S. A. Rahman, I. Song, M. K. H. Leung, I. Lee, and K. Lee, "Fast action recognition using negative space features," *Expert Systems with Applications*, vol. 41, no. 2, pp. 574-587, 2014.

[51] A. Rahman Ahad, *Computer Vision and Action Recognition: A Guide for Image Processing and Computer Vision Cummunity for Action Understanding*, 1st ed., I. Khalil, Ed. Paris, France: Atlantis Press, 2011, vol. 5.

[52] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40-79, 2010.

[53] A. D. Forbes, "Classification-algorithm evaluation: Five performance measures based onconfusion matrices," *Journal of Clinical Monitoring*, vol. 11, no. 3, pp. 189-206, 1995.

[54] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on. IEEE*, Tampa, United States, 2008, pp. 1-4.

[55] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE*

Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, 1979.

[56] S. Haykin, *Neural Networks and Learning Machines*, Third Edition ed. Prentice Hall, 2009.

[57] F. Rosenblatt, "The Perceptron – A Perceiving and Recognizing Automaton," Cornell Aeronautical Laboratory Report 85-460-1 , 1957.

[58] H. Chih-Wei and L. Chih-Jen, "A Comparison of Methods for Multiclass Support Vector Machines," *Neural Networks, IEEE Transaction on*, vol. 13, no. 2, pp. 415-425, Mar. 2002.

[59] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 12, pp. 2247-2253, 2007.

[60] I. S. P. Co., "The body and its proportions," in *Art of Drawing the Human Body*. Barcelona, Spain: Parramon, 2004, pp. 13-17.

[61] G. Bradski, "The OpenCv Library," *Doctor Dobbs Journal*, vol. 25, no. 11, pp. 120-126, Nov. 2000.

[62] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, "A Survey of Video Datasets for Human Action and Activity Recognition," *Computer Vision and Image Understanding*, vol. 117, no. 6, p. 633–659, 2013.

[63] D. Tran and A. Sorokin, "Human activity recognition with metric learning," in *Proceedings of the 10th European Conference on Computer Vision ECCV 2008*, 2008, pp. 568-561.

[64] N. Gkalelis, H. Kim, A. Hilton, N. Nikolaidis, and I. Pitas, "The i3dpost multi-view and 3d human action/interaction database," in *Visual Media Production (CVMP'09) Conference*, 2009, pp. 159-168.

[65] Z. Lin, Z. Jiang, and L. S. Davis, "Recognizing actions by shape-motion prototype trees," in *Computer Vision, 2009 IEEE 12th International Conference on* , Kyoto, 2009.

[66] A. Fathi and G. Mori, "Action recognition by learning mid-level motion features," in *Computer Vision and Pattern Recognition. CVPR 2008*, 2008, pp. 1-8.

[67] N. Ikizler and P. Duygulu, "Human action recognition using distribution of oriented rectangular patches," *Human Motion–Understanding, Modeling, Capture and Animation*, pp. 271-284, 2007.

[68] N. Gkalelis, N. Nikolaidis, and I. Pitas, "View indepedent human movement recognition from multi-view video exploiting a circular invariant posture representation," in *Multimedia and Expo. ICME 2009. IEEE International Conference on*, 2009, pp. 394-397.

[69] M. B. Holte, T. B. Moeslund, N. Nikolaidis, and I. Pitas, "3d human action recognition for multi-view camera systems," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), IEEE International Conference on*, 2011, pp. 342-349.

[70] A. Iosifidis, A. Tefas, and I. Pitas, "View-invariant action recognition based on artificial neural networks," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 23, no. 3, pp. 412-424, 2012.

[71] A. Iosifidis, A. Tefas, and I. Pitas, "Multi-view action recognition based on action volumes, fuzzy distances and cluster discriminant analysis," *Signal Processing*, vol. 93, no. 6, pp. 1445-1457, 2013.

[72] M. Meneses. (2014, Feb.) Mariomenesescpo. [Online]. http://mariomenesescpo.com

[73] WEBGATE. [Online]. http://www.webgate-usa.com

[74] S. Grinter. (2013, Oct.) IHS Technology. [Online]. https://technology.ihs.com