6-23-2018

Trajectory-based Human Action Recognition

Pejman Habashi
University of Windsor

Trajectory-based Human Action Recognition

By

Pejman Habashi

A Dissertation
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfilment of the Requirements for
the Degree of Doctor of Philosophy
at the University of Windsor

Windsor, Ontario, Canada

2018

Trajectory-based Human Action Recognition

by

Pejman Habashi

APPROVED BY:

F. Qureshi, External Examiner
University of Ontario Institute of Technology

K. Tepe
Department of Electrical and Computer Engineering

D. Wu
School of Computer Science

J. Lu
School of Computer Science

I. Ahmad
School of Computer Science

B. Boufama, Advisor
School of Computer Science

May 16, 2018

Declaration of Co-Authorship and Previous Publications

I. Co-Authorship

I hereby declare that this dissertation incorporates material that is a result of joint research, as follows: The entire dissertation and all the papers listed in the next section were written with the guidance and direct supervision of my supervisor Dr. Boubakeur Boufama. This dissertation contains materials in collaboration with Dr. Boubakeur Boufama and Dr. Imran Ahamd as well. The collaborations are covered in chapters 3, 4, 5 and 6 of the dissertation. In all chapters, the key and primary contributions, experimental designs, data analysis and interpretation, were implemented by the author of this dissertation, and the contributions of the co-authors were primarily through the provision of proof reading and reviewing the research papers regarding the technical and vocabulary contents.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my dissertation, and have obtained written permission from each of the co-author(s) to include the above material(s) in my dissertation.

I certify that, with the above qualification, this dissertation, and the research to which it refers, is the product of my own work.

II. Previous Publication

This dissertation includes four original papers that have been previously published or submitted for publication in peer reviewed journals and conferences, as follows:

Dissertation Chapter	Publication title/full citation	Publication status
Chapter 3	Pejman Habashi, Boubakeur Boufama, and Imran Shafiq Ahmad."The bag of micromovements for human activity recognition". In *International Conference Image Analysis and Recognition*, pages 269-276. Springer, 2015.	Published
Chapter 4	Boubakeur Boufama, Pejman Habashi, and Imran Shafiq Ahmad. "Trajectory-based human activity recognition from videos". In *Advanced Technologies for Signal and Image Processing (ATSIP)*, 2017 3rd International Conference on. IEEE, 2017.	Published
Chapter 5	Pejman Habashi, Boubakeur Boufama, and Imran Shafiq Ahmad. "A better trajectory shape descriptor for human activity recognition". In *Image Analysis and Recognition: 14th International Conference*, ICIAR 2017, Montreal, QC, Canada, July 57, 2017, Proceedings, pages 330-337, Cham, 2017. Springer International Publishing.	Published
Chapter 6	Pejman Habashi, Boubakeur Boufama, and Imran Shafiq Ahmad. "Disparity-Augmented Trajectories for Human Activity Recognition"	Submitted

I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my dissertation. I certify that the above material describes work completed during my registration as a graduate student at the University of Windsor.

III. General

I declare that, to the best of my knowledge, my dissertation does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my dissertation, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my dissertation. I declare that this is a true copy of my dissertation, including any final revisions, as approved by my dissertation committee and the Graduate Studies office, and that this dissertation has not been submitted for a higher degree to any other University or Institution.

Abstract

Human activity recognition has been a hot topic for some time. It has several challenges, which makes this task hard and exciting for research. The sparse representation became more popular during the past decade or so. Sparse representation methods represent a video by a set of independent features. The features used in the literature are usually low-level features. Trajectories, as middle-level features, capture the motion of the scene, which is discriminant in most cases. Trajectories have also been proven useful for aligning small neighborhoods, before calculating the traditional descriptors. In fact, the trajectory aligned descriptors show better discriminant power than the trajectory shape descriptors proposed in the literature.

However, trajectories have not been investigated thoroughly, and their full potential has not been put to the test before this work. This thesis examines trajectories, defined better trajectory shape descriptors and finally it augmented trajectories with disparity information.

This thesis formally define three different trajectory extraction methods, namely *interest point trajectories* (IP), *Lucas-Kanade based trajectories* (LK), and *Farnback optical flow based trajectories* (FB). Their discriminant power for human activity recognition task is evaluated. Our tests reveal that LK and FB can produce similar reliable results, although the FB perform a little better in particular scenarios. These experiments demonstrate which method is suitable for the future tests. The thesis also proposes a better trajectory shape descriptor, which is a superset of existing descriptors in the literature. The examination reveals the superior discriminant power of this newly introduced descriptor. Finally, the thesis proposes a method to augment the trajectories with disparity information. Disparity information is relatively easy to extract from a stereo image, and they can capture the 3D structure of the scene. This is the first time that the disparity information fused with trajectories for human activity recognition.

To test these ideas, a dataset of 27 activities performed by eleven actors is recorded and hand labelled. The tests demonstrate the discriminant power of trajectories. Namely, the proposed disparity-augmented trajectories improve the discriminant power of traditional dense trajectories by about 3.11%.

Dedication

To my beloved wife Minoo ...

Acknowledgements

No great work in the world was ever done solely by one person. Several persons helped me during my journey as a Ph.D. student. First and foremost, I would like to express my gratitude to my advisor, Dr. Boubakeur Boufama, for his support of my Ph.D. study. His motivation, patience, immense knowledge, and insights helped me a lot during these years and gave me the vision about my own research. I am also grateful for the flexibility he gave me during my study so that I could explore independently in the areas of my interest. In addition, I want to thank my co-supervisor, Dr. Imran Ahmad for his support and guidance during my research. Besides, I would like to thank the rest of my thesis committee: Dr. Dan Wu, Dr. Jianguo Lu, and Dr. Kemal Tepe for their valuable comments and encouragements. My sincere thank goes to all the professors and teachers who taught me a lot throughout my life, their wisdom opened my eyes to the beauty of science. I want to thank my parents for raising me and their encouragement to continue my education to this level. I am especially indebted and thankful to my wife, for her continuous support, sacrifice, and encouragement during these years. She is the one who endured with me all the hardships and even took twice responsibilities in the personal life so that I could focus on my studies.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
AI	Artificial Intelligence
ASL	Americal Sign Language
BOW	Bag-Of-Words
CPU	Centeral Processing Unit
CV	Computer Vision
DAT	Disparity-Augmented Trajectory
EM	Expectation Maximization
FAST	Features from Accelerated Segment Test
FB	Farnback optical flow based trajectories
FV	Fisher Vector
FVE	Fisher Vector Encoding
GHz	Gigahertz
GMM	Gaussian Mixture Model
HAR	Human Activity Recognition
HCI	Human-Computer Interaction
HMM	Hidden Markov Model

HOG	Histogram of Oriented Gradients
HOF	Histogram of Oriented Flows
IP	Interest Point trajectories
KLT	Kanade-Lucas-Tomasi
LDA	Linear Discriminant Analysis
LK	Lucas-Kanade based trajectories
MBH	Motion Boundary Histograms
MEI	Motion Energy Image
MHI	Motion History Image
MII	Motion Intensity Image
RAM	Random Access Memory
RANSAC	Random Sample Consensus
RBF	Radial Bbasis Function
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
SVM	Support Vector Machines
TDD	Trajectory-pooled Deep-convolutional Descriptors
TOF	Time Of Flight

CHAPTER 1

Introduction

1.1 Artificial Intelligence and Computer Vision

Artificial intelligence (AI) is the process of making machines to represent similar intelligence as human beings or at least pretend it[1]. Most of the work in this area, is dedicated to creation of specific domain tasks; sometimes called weak AI. For example, machine vision is the process of enabling machines to view the world similar to human beings.

Nowadays, cameras are available and used for many different tasks; from surveillance systems to smartphones and even on many gaming consoles. Cameras capture the visual world and transform it into digital data. Arguably, their task is quite similar to the human eye. A computerized system can compress, store or retrieve these data in different image or video formats[2].

Computer vision, on the other hand, is the process of analyzing digitalized images and videos to add semantical meaning to them. For example object detection aims to identify different objects in an image, or the scene categorization in which the algorithm should determine if a particular picture shoots in a beach or an office.

Computer vision usually incorporates (and sometimes enhances) the algorithms and tools from other branches of AI, namely machine learning and pattern recognition disciplines, which are specialized tools for learning from experience. The learning process is usually done by presenting many instances of different classes to the algorithm. Learning process aims to identify the type of new samples provided to it. The learning algorithms is split into two major sub-categories: supervised and unsupervised. If the instances pre-

[1] Refer to *Turing Test* and *Chinese room argument*

[2] A video here is merely a sequence of static images.

sented during learning phase has the class labels with them, it is called supervised learning. Otherwise, the algorithms are referred to as unsupervised learning. Most of the work in machine vision uses supervised machine learning. The ultimate goal of supervised learning algorithms is to create a model that generalizes the training samples to a broader model and detect the unseen samples (in addition to seen samples) correctly.

Human activity recognition is a trivial task for almost any human. It is often easy for a person to say what other people do only by watching them performing the activity. This simple task is moderately challenging for a computer vision algorithm.

1.2 Human Activity

Humans perform different tasks every day. Each of these tasks usually involves the human body movements in a certain amount. For instance, walking and running involve movement of the majority of human limbs, while typing on a computer usually solely contains movements of fingers and eyes. Some of the human body movements have particular meaning for other humans. As a result, they usually have a name. On the other hand, other movements do not have a name.

Arguably human movements can be divided into four categories based on their level of complexities:

1. Gestures
2. Actions or activities
3. Interactions
4. Group activities

The simplest form of human movements are the gestures, which are separate movements of parts of human body. They usually only contain hand movements. Hand waving and American sign language (ASL) are good examples of gestures.

Actions or activities are particular movements of a single human that usually require the whole body and often have a meaningful interpretation. In other words, there is a phrase in a natural language that can be used to describe it. For example: “walking” or “shooting a ball”. Although some authors believe that activities are more complicated than actions, others used these words interchangeably. Throughout this dissertation, we consider them the same as it is difficult to draw a line between them.

Interactions are activities that can not be done solely by a single person. Human-human and human-object interactions are two main types of interactions. Typical examples are two persons fighting or a person carrying a suitcase.

Group activities are the most complicated form of human motion. Group activity referred to a group of people, usually more than two, interacting to achieve a common objective. For example a group of people playing football or a group of people cooperating to steal a suitcase.

1.3 Problem Statement

Automatic human activity recognition (HAR) is the task of analyzing human movements in a video, especially the analysis of the whole body movement, to detect the type of activity he or she is doing. In a more specific form, it can be considered as the task of labeling unknown video clips, while each of them contains a single human performing a single activity. Here we are interested in analyzing the whole body movements.

1.4 Applications

HAR has many potential applications. Here we name four groups of application:

1. Video surveillance
2. Human-computer interaction (HCI)
3. Automatic video content analysis
4. Animation synthesis.

Video surveillance has several applications nowadays. Patient monitoring and public area surveillance help to protect human lives every day. Traditionally, cameras capture live videos from the area under the surveillance, and a group of security experts monitors these videos for potential threats. Human monitoring is expensive, and it is also prone to error. In an ideal situation, a software might replace the human surveillance totally, but it can also be used as an assistant to humans. For example, the system may draw attention to some abnormal activities in a crowded terminal by raising a red flag.

Human-computer interaction (HCI) is another potential application for human activity recognition. Human-computer interactions have been changed several times during the history: punch cards, keyboards, mice, and touchable screens are just some examples. Nowadays, there are some off-the-shelf cameras that can take pictures when people smile in front of them. Some game consoles follow player movements and respond to them accordingly. The human-like robots in the labs need to analyze human actions to communicate with them more effectively. Automatic human activity recognition technologies could lead to the design of next generations of human-computer interfaces.

Another application of HAR is the analyzing of existing videos contents. With the growth of the internet and with the help of web 2.0 technologies, there is a great deal of online digital content, including images and videos. Most of them are either not tagged or tagged poorly by human subjects. Annotating online videos can help search engines a lot, which enables people to search videos based on their content. A very sophisticated system might be able to produce a full report for a complex sports activity like tennis or soccer or even analyzes the game.

Finally, analyzing human movements from videos enables artists to produce more realistic animations or games, by synthesizing human actions based on real human activities.

1.5 Challenges

Despite the significant amount of work that has been done in this area, during the past twenty years or so, there are still little off-the-shelf products available[3]. There are many challenges in the creation of a human activity recognition system. Namely two groups of problems should be considered: natural and vision based.

Natural difficulties caused by human movement nature and can not be avoided. Here we name a few of these difficulties:

- *Definition*: There is no precise definition of human activity. Some human body movements are considered an action and have a name, while other movements do not have a name. Sometimes, even known human activities are hard to distinguish. For instance, there is no clear cut between jogging and walking, and interestingly many methods confuse these two.

- *Complexity*: Different activities may have varying amounts of complexities. For example, walking is sometimes considered a more complicated activity compared to sitting down because it contains different movements, is a cyclic activity, and usually takes longer time. Some methods perform better in the detection of cyclic actions, while others better model more straightforward activities and creating a model, that is flexible enough to handle different complexity levels, is challenging.

- *Variations in shapes and speed*: Many human activities do not have a predefined pattern and people perform them in different forms and rates. In fact, each person might have his/her unique style of doing a specific activity. For example, there are methods to identify humans using their gait [27]. A human activity recognition, on the other hand, should ignore these differences and focus on detecting human activity regardless of the person that performs it.

- *Parallelism*: People can perform activities simultaneously. For example, they can handwave while walking. It means that the created models not only should be able

[3] There are few products in the entertainment industry. Like Microsoft Kinect, which only works indoor and in certain range.

to distinguish between different activities, but also they should detect if these two activities are performing at the same time. Few researchers have focused on detection of simultaneous events, for example [28, 29, 30].

Different sensors can be used for HAR [31]. This dissertation focuses on vision-based HAR. Vision has its own challenges, so when a camera is used as the primary sensor, these challenges should be taken into account.

- *Background, color, and texture*: The input of a vision based system is a video, which contains a lot of irrelevant information. The background, colors, and textures are not directly related to the activity in the video and should be eliminated. Markers on the human body, background subtraction, and human silhouettes are some of the methods used to remove unwanted details.

- *Human shapes and scales*: Humans have different shapes (big, thin, tall, etc.) and their relative distance to the camera can cause them to appear smaller or larger in the video. This kind of information is irrelevant to HAR. Other methods, like sparse representations and human skeletons, can remove this unrelated visual information as well [32].

- *Change of view point*: When the angle between the camera and human subject changes, the visual information extracted from the video also changes. The visual clues, which are usually used for human activity recognition, will vary by the shift in the viewpoint [32].

- *Clutter*: Clutter exists in many different computer vision applications. Few methods can cope with temporal clutter like [12]. There are even fewer methods that can work when the partial human body is visible. Sparse representations seem robust to this issue, but most of the time, extracted clues are not enough for a reliable HAR.

- *Multiple humans*: Many proposed methods in the literature assume that there is only one activity of interest in the field of view, and many of them bear no other human (even no other movement) in the video. The presence of multiple humans, doing arbitrary different activities, makes the process harder. Even though some methods [15, 30] can handle this kind of situations, but those approaches need more computational resources.

1.6 Contributions

In an exciting experiment, Johansson et al. [2] attached between five and thirteen markers to the main human joints and recorded videos from humans doing different activities in front of a camera. The cameras recorded only the movements of these markers. The footage of these white points moving on a black screen was shown to different groups of people. Subjects could tell that the markers are attached to the human body, and almost in all cases, they could tell what activity was done by the actor.

It is still unclear if human brain uses only the two dimensional (2D) location of these points for classification or it creates a three dimensional (3D) model of the points during the recognition process. It is hard to answer this question directly, but this dissertation shows that using some 3D data can improve the classification results[4](See chapter6 for details). The cameras project 3D space to 2D space and the direct analysis of images produces 2D data. To map a 2D point to 3D, at least two views of the same point are required. Then multiple view geometry can map these 2D points to 3D.

The main contributions of this dissertation are as follows:

- Trajectories for human activity recognition: Although the use of trajectories for human activity recognition is not entirely new, we have shown that trajectory shapes can be beneficial for human activity recognition. In particular, we have formally defined three different trajectory extraction algorithms and compared their discriminant power with each other. The trajectory extraction is the first step in extraction of appearance-based descriptors. As a result, trajectories can decrease the computation time compared to traditional trajectory-aligned appearance-based methods. This work is published in the paper [33] and explained in Chapter 4.

[4] At least in certain scenarios

- Better trajectory shape descriptors for human activity recognition: This dissertation proposes a new trajectory shape encoding algorithm, which is a superset of existing trajectory shape extraction algorithms. It demonstrates that the current methods in the literature usually capture only the velocity of trajectories, our proposed method capture acceleration and higher order information in addition to the velocity of the motion. Our tests confirm that this method can improve the classification results. This contribution is published in [34] and explained in Chapter 5.

- Disparity-augmented trajectories for human activity recognition: Disparity can be calculated for a point with at least two views, and it can be translated into the depth of field. Disparity is not the same as 3D data, but it has beneficial 3D information. Disparity information is easier to extract compared to 3D data. For a complete Euclidean 3D reconstruction of a scene, the camera parameters are required, which are hard or impossible to extract in certain scenarios. Although the full 3D Euclidean reconstruction of the scene can be beneficial for human activity recognition, it might be unnecessary. This research proposes to use disparity information in addition to 2D trajectory information for human activity recognition. The method and results are explained in Chapter 6.

- Stereo dataset for HAR: To demonstrate the effectiveness of the proposed method, we have created a stereo dataset for human activity recognition. This dataset has a total of 27 different activities performed by eleven volunteers in different scenarios. The details of proposed dataset are explained in Chapter 6.

1.7 Organization

Chapter 2 reviews the state of the art in the area. Chapter 3 proposes the *micromovements* for human activity recognition. Micromovements are a particular form of trajectories, and a multi-view setting can be used to augment micromovements with disparity data. This chapter introduces and examines the idea of using stereo vision to improve the accuracy of a human activity recognition system. Chapter 4 compares different trajectory extraction algorithms for human activity recognition process and shows the pros and cons of trajectories. Chapter 5 proposes a better trajectory shape encodings, which is the process of encoding the shape of motion into a metric space in a way that the similar trajectories are close on that space. The chapter also compares the proposed method with traditional trajectory shape encoding algorithms, and shows the effectiveness of the approach. Chapter 6 proposes to augment the trajectories with disparity information. The latter can reflect the depth of field, and adds additional information to 2D trajectories. That chapter shows that disparity-augmented trajectories can outperform the 2D trajectories with a good margin. It also explains a new stereo dataset created for human activity recognition. A conclusion are provided in Chapter 7.

CHAPTER 2

Previous Works

2.1 Introduction

This chapter reviews some of the existing methods for human activity recognition in the literature. Many different approaches for human activity recognition have been proposed and tested in the literature. In this chapter, we focused on vision-based methods which target the identification of single human performing single task.

Some methods in the literature are trying to solve the HAR problem hierarchically. [32] coined and explained hierarchical approaches versus single layered approaches. Single-layered approaches attempt to detect activities, directly from the set of features extracted from the video. On the other hand, hierarchical methods, break down the movement into several smaller/simpler actions, each of which may be split into even simpler sub-activities. At the bottom line, there are atomic activities, which are simple activities that can not be broken into simpler forms. Most of the hierarchical approaches used single layered methods for detecting these atomic activities. This chapter focuses on the single-layered approaches only.

This chapter is organized as follows. Section 2.2 puts existing methods into five different categories. As there are many methods proposed in the literature, we have a quick overview over them. One of this categories are sparse representation methods, which has been more popular recently and the methods proposed in this dissertation can be considered a subset of these methods. Therefore a more detailed overview of the sparse representation methods provided in Section 2.2.5. Section 2.3 discusses trajectory-based methods (the most relevant work) with a lot of details. Finally, Section 2.4 describes two popular methods used in the lierature to prepare data for learning sparse features.

2.2 Representation Methods

Feature extraction is the core of any machine learning system. The feature extraction should eliminate irrelevant details while keeping the main discriminative information. The input of human activity recognition system is the video(s) of human subjects. These videos should be converted into vectors in a metric space so that they can be discriminated by an existing machine learning method.

Based on the way that different methods extracted the features, we put them in five different categories:

1. *Human skeleton* methods are the ones that extract human skeleton before the extraction of features. The human skeleton might be represented in two or three-dimensional space.

2. *Body parts tracking* methods tried to detect and follow the human body parts in the video.

3. *Silhouette based* approaches try to represent the activity as a single image.

4. *Actions as 3D objects* refers to the methods that concatenate the silhouettes and tries to create a 3D shape of action.

5. *Sparse Representation* approaches which represents a video by a set of local features.

Each of these methods are explained in its own section.

2.2.1 Human Skeleton Based Methods

The work of Johansson [2] inspired many researchers to use joint locations for HAR. He attached between 5 and 13 labels to the human body and recorded the movement of these labels with a camera (Figure 2.2.1). He showed that all human subjects could tell that the moving points are attached to the human body, and they can name the activity which was performing. It is not clear if humans make a 3D model in their mind or just use 2D information in the recognition process.

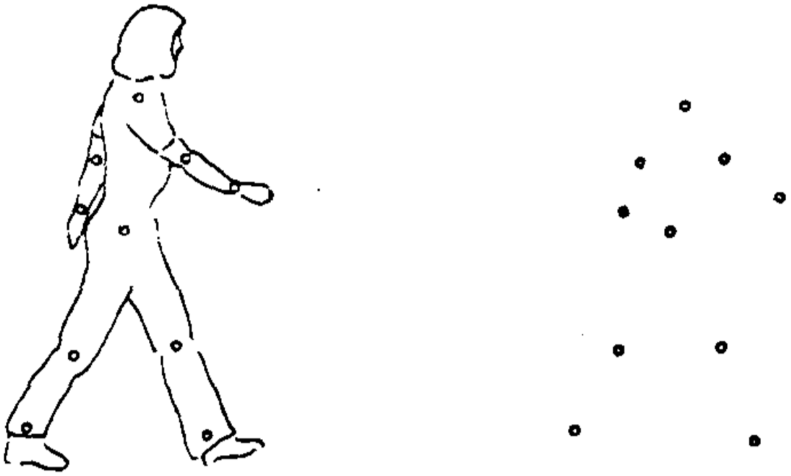

Figure 2.2.1: Johansson's joint model [2]

Human Skeleton Extraction

The first step and the main challenge of human skeleton based methods, is the estimation of human joint locations. Some early methods rely mostly on special motion capture equipment (like MoCap sensor) or special markers on human body [4, 7, 35]. Human skeleton extraction is still challenging, but there are active cameras (like Microsoft Kinect) that can facilitate this step.

Depth cameras produce new opportunities in machine vision in general and human activity recognition in particular. For example Shotton et al. [3] consider each part of human body as a 2D object (Figure 2.2.2) and by using randomized decision forests, each pixel of the image is assigned to different classes with a different probability. These probabilities demonstrate the chance of a particular pixel belonging to a different body part. By projecting this information back to the depth image, several hypotheses' about the joint locations are calculated, and the mean shift algorithm is used to assign a confidence measure to each one of these hypotheses.

In another attempt, Uddin et al. [5] have used stereo vision cameras and the method which was proposed in [36], for extraction of disparity image. Later triangulation was used to calculate depth. They have used object tracking and face detection to locate person's body, face, torso, and hand positions. They co-registered their model to the 3D data using expectation maximization (EM) algorithm.

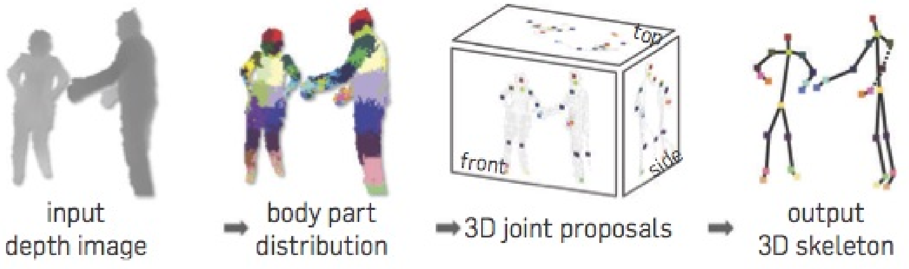

Figure 2.2.2: Skeleton extraction proposed in [3]

3D Joint Methods

These methods use 3D joint location information for human activity recognition. One of these attempts done by Campbell and Bobick [4], who attached 14 markers to the human body and a commercial system found the location of them in the 3D space (Figure 2.2.3). They defined a body phase space, in which, each dimension represents one of the human posture independent parameters (e.g., each joint position or angle). Now, each human pose is mapped to a point, and every human activity is represented by a curve in this space. These curves are then remapped to a 2D subspace, where the distance of the unknown posture (a point in this space) with the curve measured to identify different activities.

In another attempt, Uddin et al. [5] designed a human kinematic with fourteen body segments and nine joints with 24 degrees of freedom (Figure 2.2.4). As a result, 24 parameters can describe a posture of this human kinematic. They used stereo vision to map the human body into this kinematic model. To achieve better discrimination, the 24 parameters of the model are mapped to another space using linear discriminant analysis (LDA). Later, these feature vectors are clustered into different codebooks. The cluster number was used as the input of a hidden Markov model (HMM), which performs the final classification of activities.

In another effort, Xia et al. [6] utilized a spherical coordinate system to make their descriptor. First, the spherical coordinate system is attached to the person's hip (Figure 2.2.5). This spherical space is split into several bins based on different angles. The radial distance is ignored to remove the scale effect. Then, probabilistic voting is used to smooth the joint location estimations and to improve robustness. LDA and K-means are used to cluster these

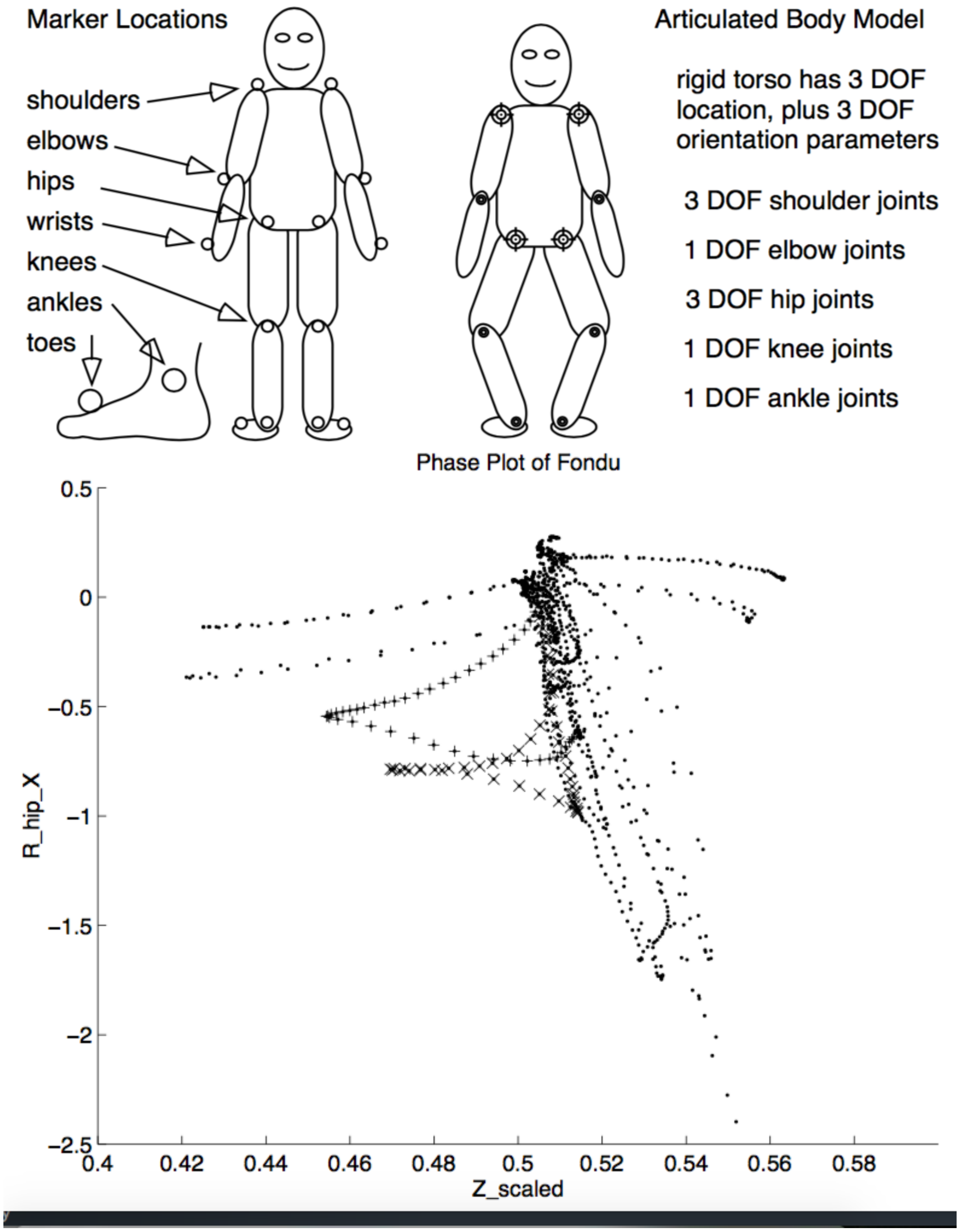

Figure 2.2.3: 3D joint models proposed in [4]

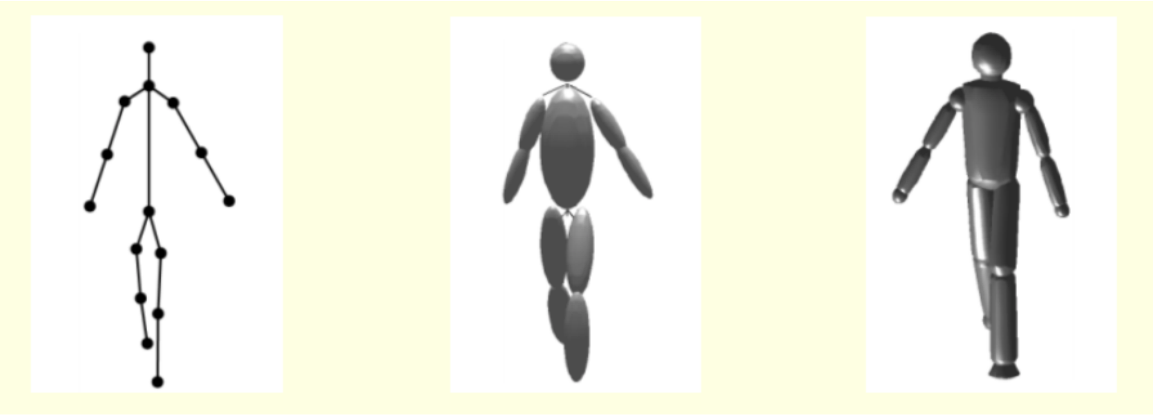

Figure 2.2.4: Human kinematic proposed in [5]

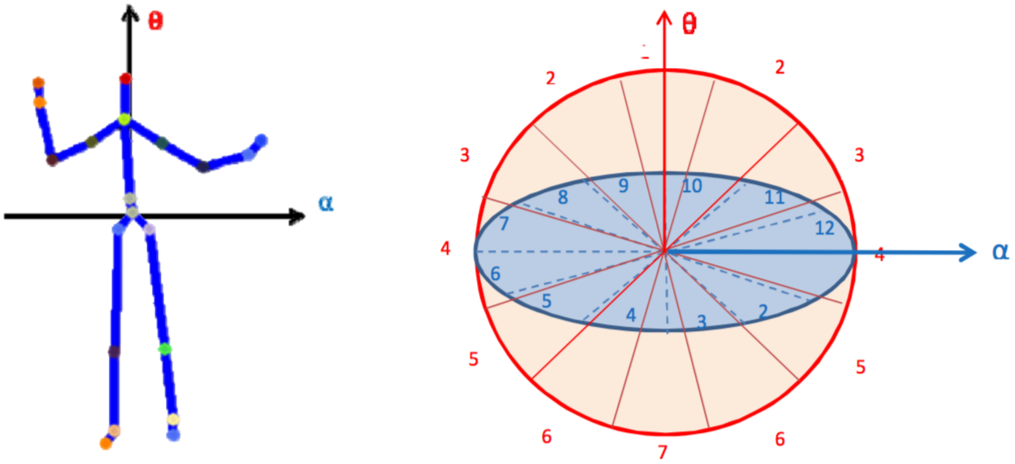

Figure 2.2.5: Spherical coordinate system aligned with human hip [6]

histograms into visual words. This way, each activity is represented as a series of observed words. Finally, an HMM is used for clustering these movements.

Barnachon et al. [37, 38] used MoCap [39] to capture the 3D skeleton of human body and create a list of poses. Then, the Hausdorff distance is used for binning the pose space and creating a histogram of poses. This histogram determines which poses are more likely for each activity. Then a cumulative sum of these histograms is used to calculate the integral histogram, which represents the likelihood of an action over time. The Bhattacharyya distance gives the similarity of two cumulative histograms. These integral histograms broke into several sub-histograms, which are then employed to learn an HMM model for online detection of activities.

2D Joint

Another group of methods used is the 2D joint location for HAR. For instance, Sheikh et al. [7] used motion of 13 landmarks of human body in video (see Figure 2.2.6) for activity recognition. Each video is summarized in a matrix of joint positions. Then each activity is considered as a linear combination of spatiotemporal action bases. By assuming an affine transformation between word and image coordinate system, a matrix for each activity in dataset is created. Each of these matrices represents a subspace and the angle between these subspaces is used for recognition. This model needs a lot of learning samples to cover the whole action space.

Another method proposed by Yilmaz and Shah [35] focused on solving the camera movement problem. 13 landmarks are attached to the human body, and the 2D location of these marks is recorded in the video. The multi-view geometry is incorporated to create a temporal fundamental matrix. The camera motion and subject motion are modeled separately. The reconstruction error used as a distance measure for activity recognition.

Human joint information is compelling in describing human movement, and it can be used for analyzing human activity. This information may be the joint location, angles between joints, the rate of change in position or angle, or join trajectories. Most of the methods that used the human skeleton are view-invariant. On the downside, these methods need special equipment to extract joint locations, and the accuracy of these methods is affected directly by the accuracy of joints location estimations.

2.2.2 Body Part Tracking Methods

Although human skeleton based methods are among first solutions proposed, the final results are not satisfactory for most cases. Another challenge for skeleton-based methods is the extraction of the human skeleton. The performance of skeleton based methods highly depend on the accuracy of the human skeleton extraction method.

Another group of methods in the literature tried to solve the HAR problem with the detection of human part locations instead of joint location.

For example, Rao and Shah [8] used hand trajectories for specific domain activity

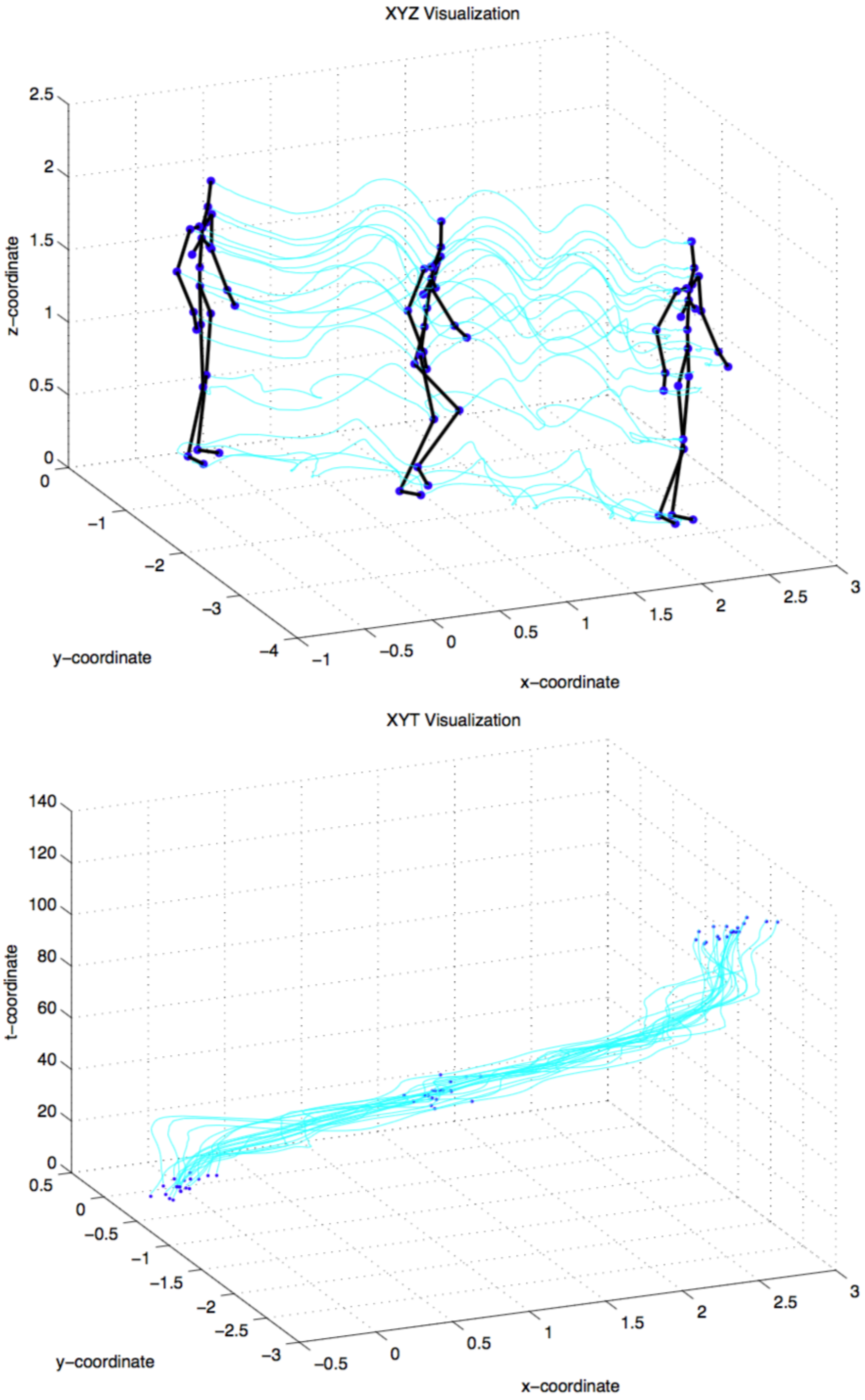

Figure 2.2.6: Representation of an activity as a linear combination of action bases [7]

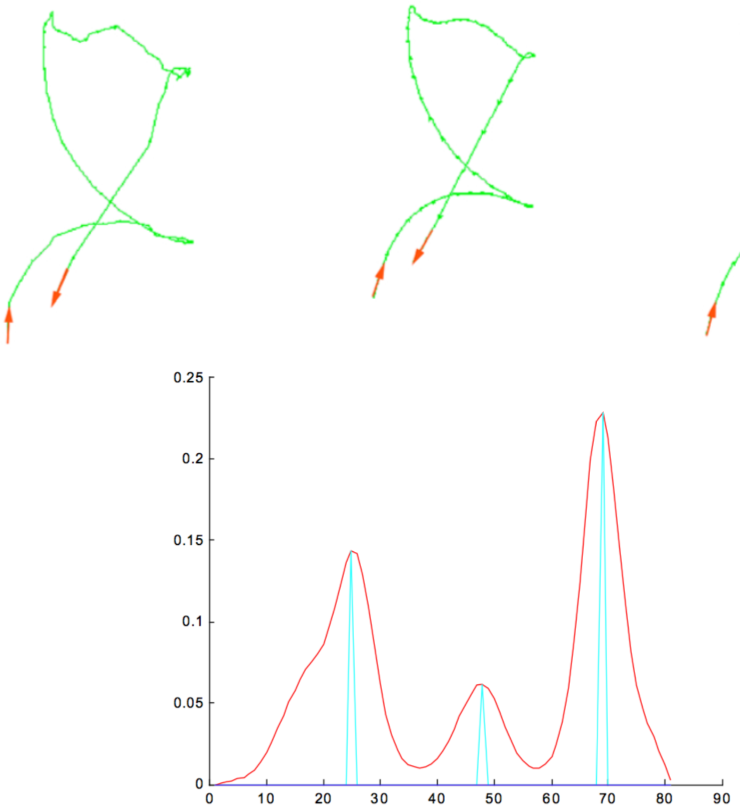

Figure 2.2.7: Hand trajectories and their flattened representation [8]

recognition. The skin area is detected by using lookup-tables and connected component analysis. Then the 2D trajectory of hand movements is created and flattened (Figure 2.2.7). These trajectories are analyzed for speed, direction, acceleration, and curvature. This information is used with a rank-nullity theorem based method for activity recognition. Although the authors prove that some specific information extracted from trajectories are view-invariant, this information is not enough for activity recognition.

The body part tracking has useful information for human activity recognition, but these methods also suffer from the shortcomings of skeleton based methods. Some methods that proposed to track face and hand can not collect enough discriminant information for a reliable human activity recognition.

2.2.3 Silhouette Based Methods

Another solution proposed to avoid human skeleton extraction was the use of human silhouettes. Human silhouettes can be extracted easily from stationary cameras, and they can remove much irrelevant and misleading information such as background, color, and the texture. Besides, silhouettes extraction does not need multiple camera or specialized equipment.

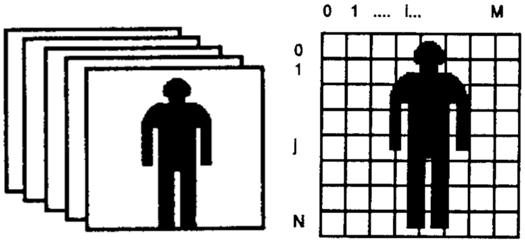

Figure 2.2.8: Silhouettes sequence and the grid used for HAR [9]

Yamoto et al. [9] introduced the use of silhouettes for activity recognition. A mesh put on the human silhouette and the ratio of black pixels to white pixels in each cell is calculated (Figure 2.2.8). Quantization of these values forms a descriptor for each frame. An HMM is used to make a model for activity recognition. This method can produce relatively good result in some tests, but it is sensitive to the shape of the actors. When different actors are used for training and testing, the performance is decreased.

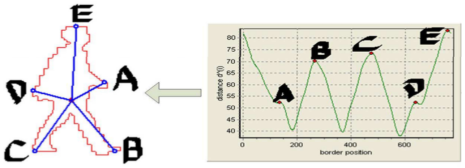

Figure 2.2.9: Star skeleton extraction [10]

In a more successful attempt, Chen et al. [10] proposed to make star skeletons from the extracted silhouettes. After extracting human silhouette, human boundary are extracted and flattened (see Figure 2.2.9). After smoothing it with low pass filter, the five extremes should represent human head and four limb positions. By connecting the center of the silhouette to the extremes a star skeleton is created. They used vector quantization to map each skeleton to a state. This way, each activity is represented as a sequence of states. Finally, the HMM is used to create the model of activity.

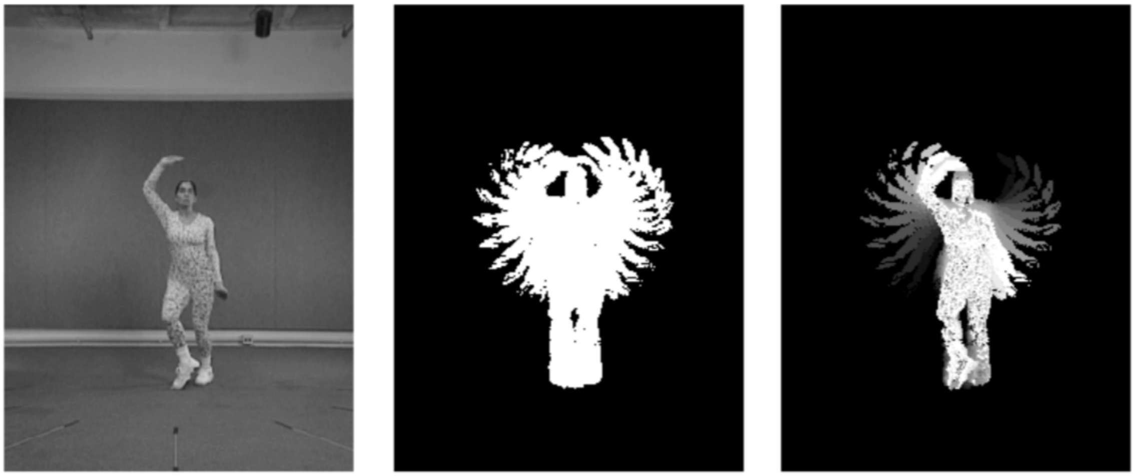

Figure 2.2.10: MHI (center) and MEI (right) [11]

Bobick and Davis [11] proposed to use human silhouettes to make *temporal templates*, which contain only 2D information. Namely, they have extracted the *motion history image* (MHI) and the *motion energy image* (MEI) (see Figure 2.2.10). MHI and MEI is the average and weighted average of consecutive silhouettes images, respectively. This way, each activity can be compressed in time and be represented by two 2D images. These images show where and how a human motion is going on. The authors extracted seven hu-moments [40] for shape analysis. They have defined a descriptor on that moments and used it for matching.

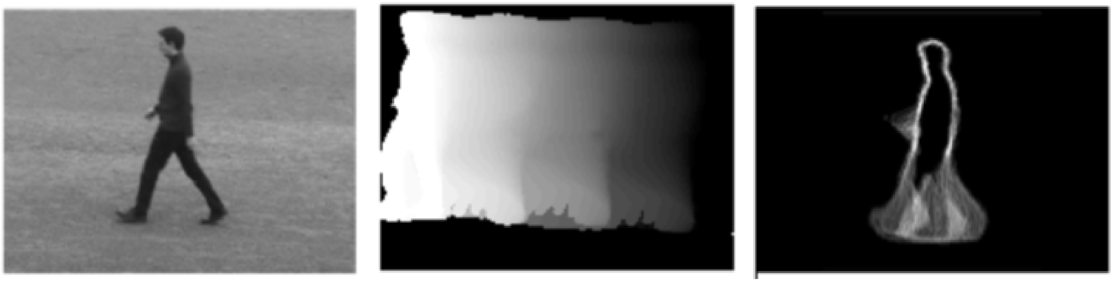

Figure 2.2.11: MHI (center) and MII (right) [12]

In order to calculate MHI/MEI, a decaying parameter and a time instance need to be defined. These parameters control the temporal fading effect of MHI and, therefore, are depended on the speed in which the action is going on. Also, MHI will change drastically in the existence of temporary occlusion. To address these issues, Diaf [12] proposed a similar representation called motion intensity image (MII) (see Figure 2.2.11). In his work, after extracting human silhouette, the center of each silhouette is found and aligned before calculating MHI. This eliminates the position information and makes the algorithm more robust.

Using MHI, MEI, and MII have many advantages. It is easy to extract and store these features. There are near real-time implementations for these methods. These methods eliminate the time factor, hence the temporal complexities are deleted, and as a result, much simpler learning methods can be used for action recognition. Silhouettes are robust to lighting and clothing conditions as well [15]. Neglecting temporal information, on the other hand, made these methods suitable for short-time activities, and they can not model complex actions. Another issue with these methods is that silhouettes are view-dependent, size-dependent and shape-dependent (e.g., big or thin person). Besides, most of silhouette extraction algorithms, assume that camera has no motion and it is static.

2.2.4 Actions as 3D objects

Silhouette-based methods showed excellent performance, especially in MHI, MEI, and MII, but these methods ignore the time factor, and so they are unable to model complex activities. Instead of calculating the mean of silhouettes as an image, some research attempts proposed to make a 3D object by stacking up the silhouettes in the spatiotemporal domain and creating a 3D object of each activity. Later, 3D shape matching is used for activity recognition.

For example, Yilmaz and Shah [13] considered each activity as a 3D object in the 3D spatiotemporal space. First, the boundaries of human body are extracted, then by relating the points on the boundary of consecutive frames, a 3D sketch or 3D object is created. This object is regarded as a rigid object (Figure 2.2.12). Action descriptors are defined as a set of straight, convex, and concave contours.

Finding point correspondence between adjacent frames is a very time-consuming task.

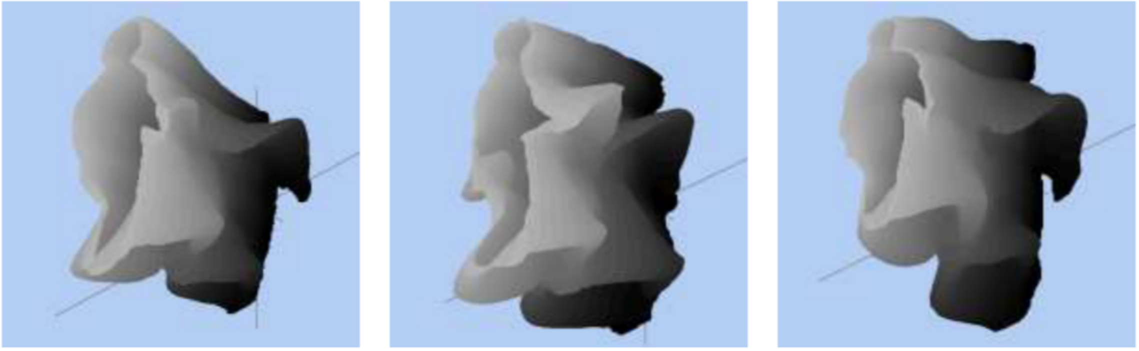

Figure 2.2.12: Action sketch [13]

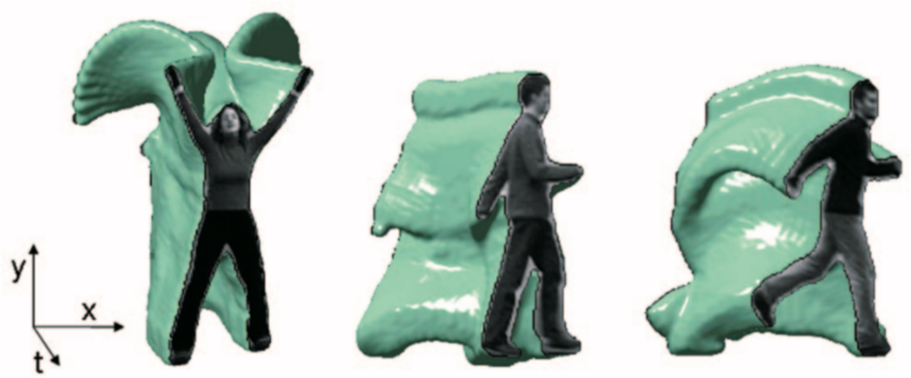

Figure 2.2.13: 3D shape of an action [14]

To solve this issue, Blank et al. [14] proposed a method for 2D shape matching using Poisson's equation. After the extraction of the 3D object of the human silhouette, the solutions of the Poisson's equation are used to define the new descriptors, which used for classification.

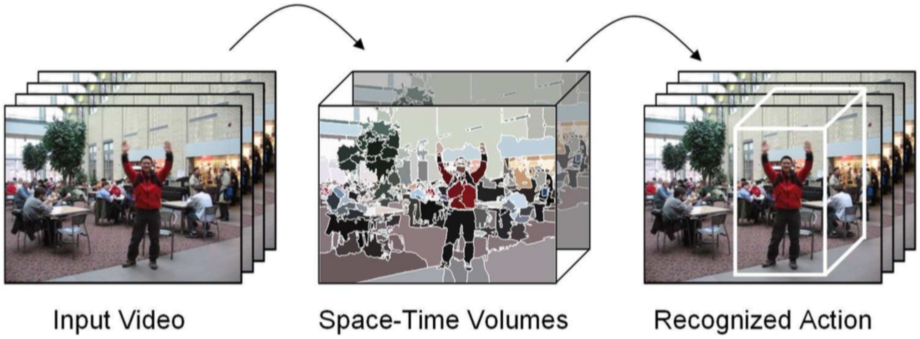

Figure 2.2.14: Super-voxels used for HAR [15]

Extracting the 3D XYT human action object is not always possible. Ke et al. [15] proposed that this object may not be even needed for activity recognition. Instead, they suggested extracting super-voxel shapes, which are 3D shapes in spatiotemporal domain. These 3D shapes boundaries should align with object boundaries. By using a similarity measure for 3D shapes, these 3D objects can be classified into different activity classes. This method is very CPU intensive, and it can not provide good performance for simple datasets (e.g., KTH).

Representing action as a 3D object is more descriptive compared to MHI and MEI because it contains temporal information as well. Although these methods are not view-invariant, they are robust to change of viewpoint, and these methods usually better handle the low-resolution images. On the down side, these methods are view dependant and are not flexible enough to cope with different variations exists in the human movements.

2.2.5 Sparse Representations

Sparse representation is one of the most popular methods in HAR. The idea is to represent each activity by a set of independent features which reappear in different instances of the activity. This way, each video is converted to a set of features.

Three kinds of sparse features are used in the literature: low-level, mid-level and high level. Low-level features are mostly appearance-based features. These kinds of features are extracted directly from pixel grayscale values. High-level features are features that are obtained from high-level representations like the human skeleton. Mid-Level features are features that are derived from other representations like trajectories. The researchers investigated low-level features thoroughly during the past decade or so. On the other hand, high level and mid-level features are more recent, and there are fewer works in this area.

Sparse Low-Level Features

These features are mostly appearance-based features and are normally extracted from the spatiotemporal representation of the video. Two main problems should be addressed in this regard, which pixels in spatiotemporal space are the most informative points and how to represent them. Feature points, by definition, are the most informative points in spatiotemporal volume, and descriptors are their signatures.

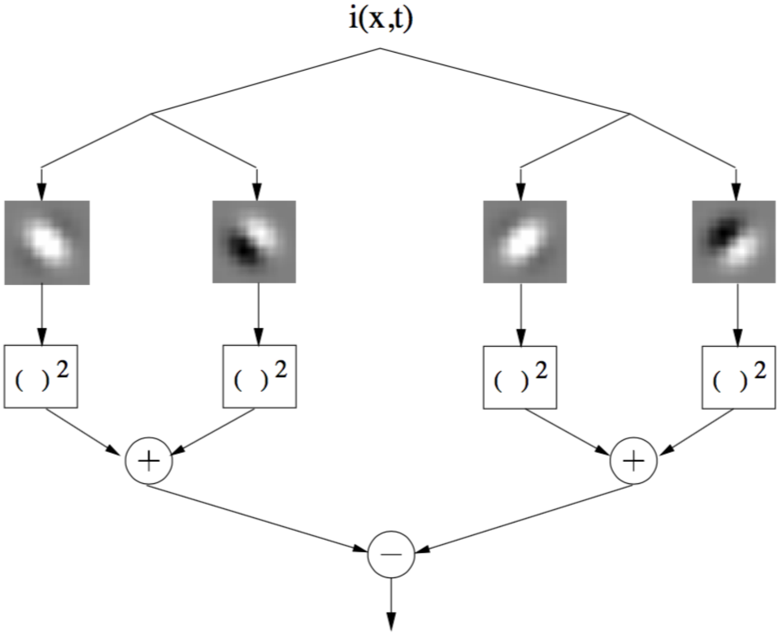

Figure 2.2.15: Gabor filters used for defining a vision based local descriptor [16]

Feature point Detector Appearance-based feature points are defined as the most informative points in the spatiotemporal space of a video. In the 2D image processing paradigm, it is shown that edges carry more information than homogeneous areas and corners carry even more information. 3D corners in spatiotemporal space are defined as spatial corners which change the direction of their movement in the temporal domain.

One of the earliest works which used appearance-based features for activity recognition is the work of Chomat and Crowley [16]. They proposed to use a Gabor filter bank to calculate scale invariant local spatiotemporal appearance-based features (Figure 2.2.15), which later passed into a multidimensional histogram to create probability density function for each activity. Activity recognition is done by using Bayes rule.

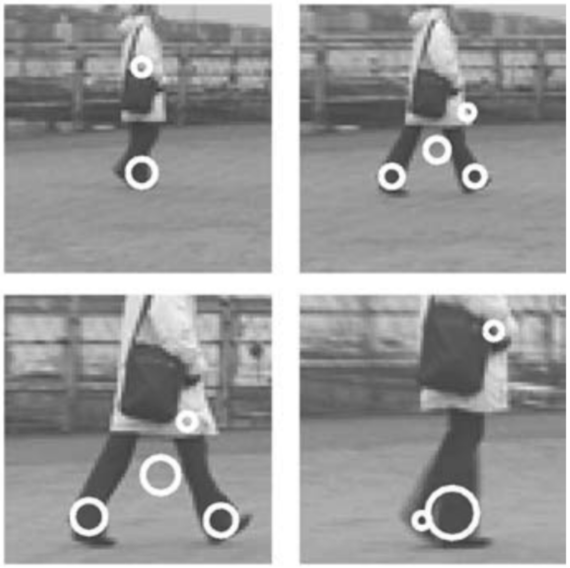

Figure 2.2.16: Sample of feature points extracted in [17]. The circles represent the location of extracted corners.

Later, Laptev and Lindeberg [17, 41] extended the Harris interest point extraction operators to the 3D spatiotemporal space and proposed a spatial and temporal scale invariant feature extraction method. Their Harris interest point detector uses the second-moment matrix to find spatial locations in a 2D image with high variation in both directions (corners) and significant variations in spatial domain. In other words, the corners of the spatial domain with a change in the direction of movement selected (see Figure 2.2.16). Then, a Gaussian kernel is used to capture the neighborhood signature. K-means clustering is used to group interest points with similar background. They propose a matching method based on the distance between these clusters.

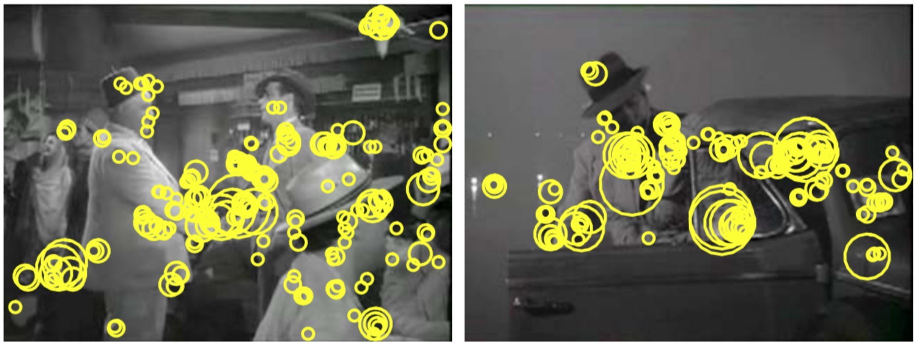

Figure 2.2.17: Sample of feature points extracted from different scales in [18]

In a separate work, Laptev et al. [18] proposed to use dense scale sampling. Accordingly, different scales are used to extract the points of interest. Histogram of Gaussian (HOG) and histogram of flow (HOF) are used as the descriptor. They concluded that HOG is more robust than HOF. A sample of their extracted feature points is visible in Figure 2.2.17.

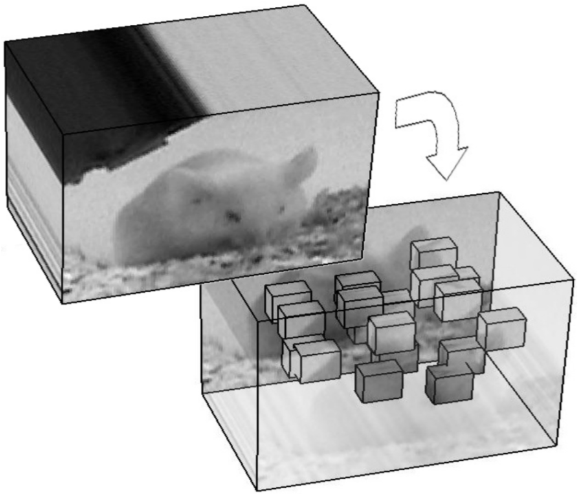

Figure 2.2.18: Samples of feature points extracted in [19]

The spatiotemporal corners are very robust feature points, but they are rare in video. Besides, these interest points do not appear in all kinds of human activity video. Dollar et al. [19] noticed these limitations and introduced another local feature point extractor named cuboid (Figure 2.2.18). The cuboid feature points are extracted by using a response function. The authors believed that cuboid locations and types should be sufficient for many recognition tasks. The response function is defined as:

$$R = (I * g * h_{ev})^2 + (I * g * h_{od})^2 \tag{2.2.1}$$

$$h_{ev}(t, \tau, \omega) = -\cos(2\pi t\omega)e^{-t^2/\tau^2} \tag{2.2.2}$$

$$h_{od}(t, \tau, \omega) = -\sin(2\pi t\omega)e^{-t^2/\tau^2} \tag{2.2.3}$$

First, the image is convolved with g, a Gaussian smoothing function in the spatial domain, then the results convolved with h_{ev} and h_{od}, in the temporal domain. These functions produce a strong response in locations with spatially distinguishing characteristics. These functions generate a lot of interest points, which are clustered with K-means clustering algorithm into few different types. Typically each activity has its own set of these types. A prototype calculated for each cluster, which represented that cluster cuboid type. Consequently, each activity is represented by a histogram of these prototypes. Matching is performed by Euclidean and χ^2 distance measures.

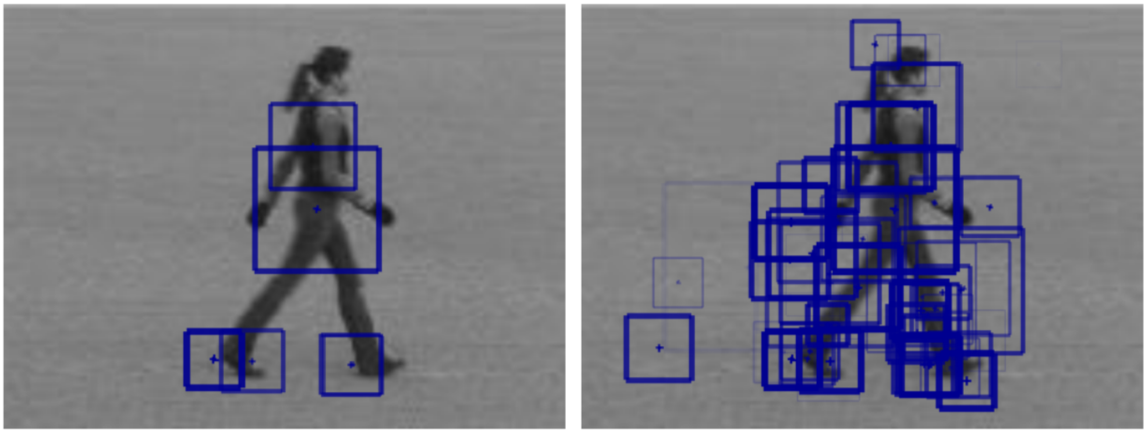

Figure 2.2.19: Sample of feature points extracted in [20]

In another attempt, Willems et al. [20] proposed a more efficient feature detector based on cuboids. They extended the existing Hessian saliency measure into the spatiotemporal domain. The integral image and better implementation details, lead to a faster feature detector (Figure 2.2.19). The speeded up robust features (SURF) is used to encode their feature points.

Even though feature points extraction algorithms are popular, Scovanner et al. [23]

showed that random sampling could produce a comparable result to these algorithms. They have extended 2D SIFT descriptor to 3D, by treating temporal domain the same as spatial domain. They have used random sampling of points for feature points.

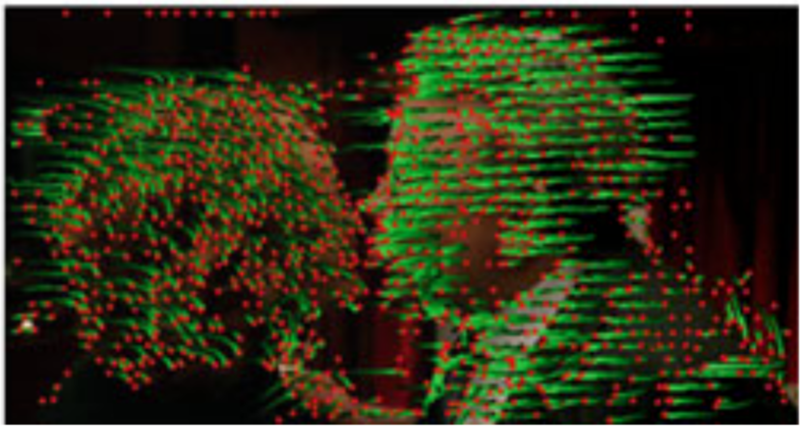

Figure 2.2.20: Sample of feature points extracted in [21]

Wang et al. [21] pointed out the differences between spatial and temporal domain and proposed to treat them accordingly. They offered to use optical flow field to track densely sampled interest points in time. Since the tracking of homogenous areas in the video is impractical, the points with low texture are detected and removed before the tracking (Figure 2.2.20). The neighborhood around these tracked feature points is stacked, and then HOG, HOF and motion boundary histograms (MBH) are calculated. MBH neutralizes the camera motion effect. In static camera scenarios, MBH and HOF produced similar results. HOG yielded better results for sports activities. Apparently, the HOG takes advantage of the background information, as the background for different sports activities is different, HOG could outperform other methods.

Camera movement produces unwanted trajectories (Figure 2.2.21). Wang and Schmid [22] proposed a model to capture the camera movement and remove unwanted trajectories. They used SURF features, which are robust to motion blur and optical flow. They assumed that there is a homography that can describe the camera motion between two consecutive frames, this excludes any independent moving object. This homography is estimated and used to remove the trajectories that are similar to the camera motion.

Feature point Descriptors After finding the interesting points in spatiotemporal space,

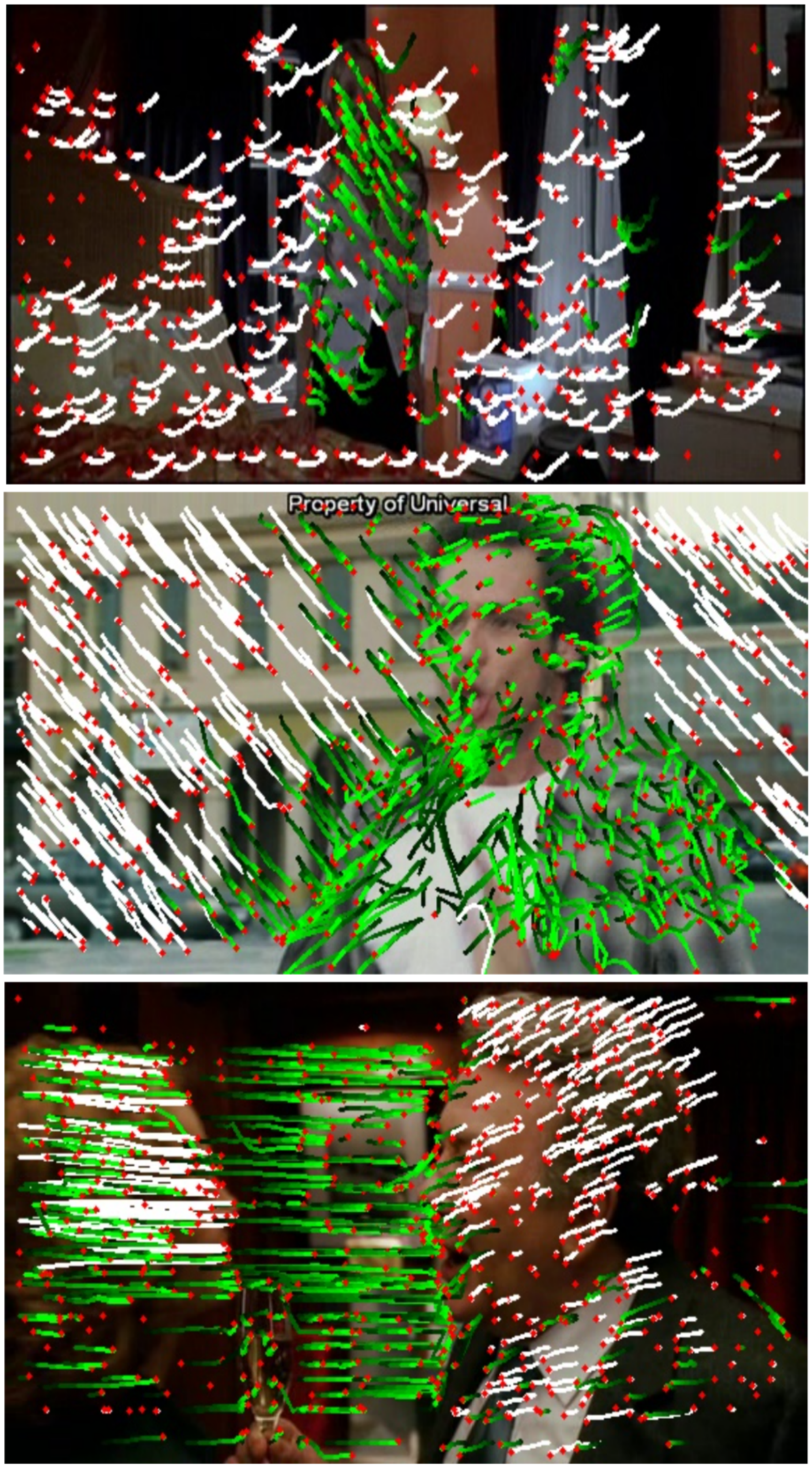

Figure 2.2.21: Unwanted trajectories produced by camera movement [22]

each of them should be represented by a signature. Many different ways to define a local signature are proposed in the literature. Here we name a few popular methods.

Histogram of Oriented Gradient (HOG) and Histogram of Flow (HOF), are two commonly used feature descriptors in HAR domain. Inspired by the success of Gradient in calculating SIFT descriptors, Dalal et al. [42] proposed HOG for images. HOG captures the visual appearance features while HOF captures the dynamic of an activity. To calculate HOG, first, a dense grid is put around each interest point. Then a local histogram of orientation for each cell is calculated, and by combining these Histograms, a 1-D descriptor is made. A contrast-normalization is typically done on overlapping blocks to make the descriptor more robust. HOF is very similar to HOG but, instead of gradient information, it encodes optical flow information.

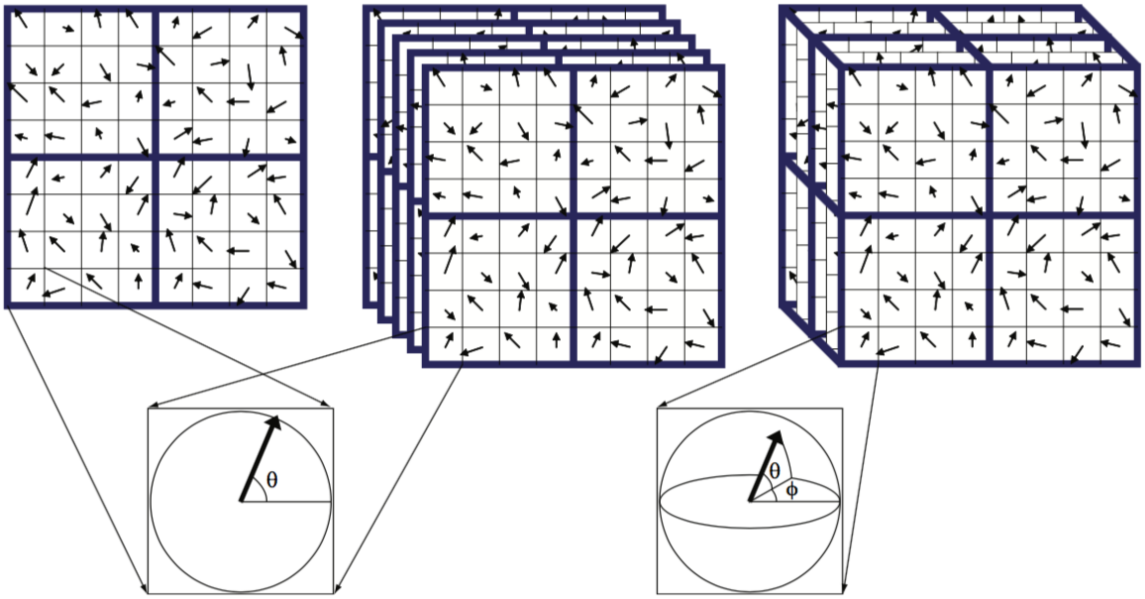

Figure 2.2.22: 2D-SIFT versus 3D-SIFT descriptor [23]

By treating the temporal domain the same as spatial domain, Scovanner et al. [23] defined a 3D-SIFT descriptor which is the extension of well known SIFT descriptors [43]. This descriptor is identical to 2D-SIFT, except there is one more angle for the gradient direction (Figure 2.2.22). The descriptor is calculated as follows: A neighborhood direction is assigned to each interest point. Later, this direction is described as two angles in 3D spatiotemporal space. After rotating the neighborhood to align with this direction, a 3D grid is put around the point of interest, and finally, for each cell, a histogram of Gaussian is created.

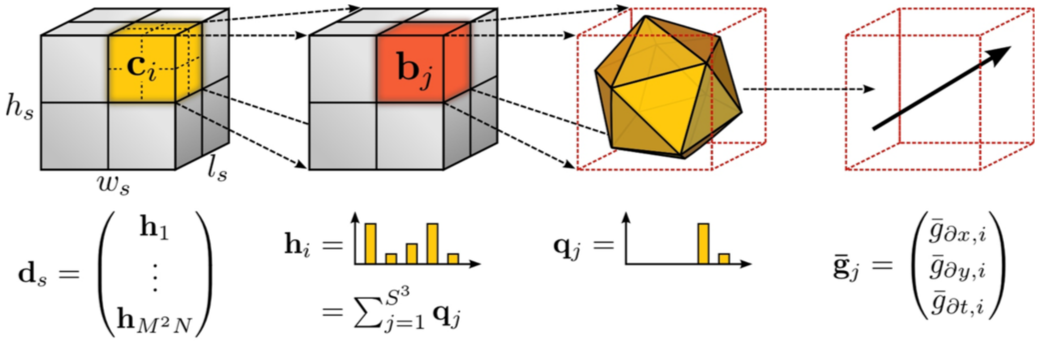
$\mathbf{c}_i$
$\mathbf{b}_j$
h_s
w_s
l_s
$\mathbf{d}_s = \begin{pmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{M^2N} \end{pmatrix}$
$\mathbf{h}_i =$
$= \sum_{j=1}^{S^3} \mathbf{q}_j$
$\mathbf{q}_j =$
$\bar{\mathbf{g}}_j = \begin{pmatrix} \bar{g}_{\partial x,i} \\ \bar{g}_{\partial y,i} \\ \bar{g}_{\partial t,i} \end{pmatrix}$

Figure 2.2.23: Quantizing the 3D direction as proposed in [24]

Inspired by the success of HOG in describing 2D still images domain, Klaser and Marszalek [24] proposed HOG3D, which is the extension of HOG to 3D. In their method, a 3D grid put around the interest-point and the mean gradient for each pixel is calculated. A dodecahedron (12-sided) and an icosahedron (20-sided) was used to quantize the directions and creating a histogram of direction (Figure 2.2.23).

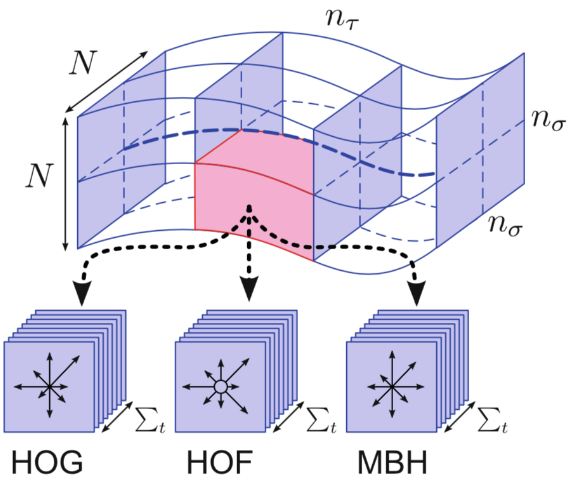

Figure 2.2.24: Aligning neighbourhood before calculating descriptor [22]

In an inspiring work, Wang and Schmid [22] proposed to use tracking information to improve local feature descriptors. They have argued that the movement of interest points may result in a poor descriptor. For example, the same point of interest, which undergoes different motion, may result in different descriptors. In their proposed method, each point of interest is tracked. They aligned the neighborhood based on tracking information and only then calculated the HOG, HOF and MBH descriptors. Figure 2.2.24 illustrates this idea. The authors showed that the descriptors, calculated after aligning the neighborhood, are more robust.

Sparse Mid-Level Features

Middle-level sparse features refer to features that are not directly extracted from pixel values, but they have no other high-level interpretations as well.

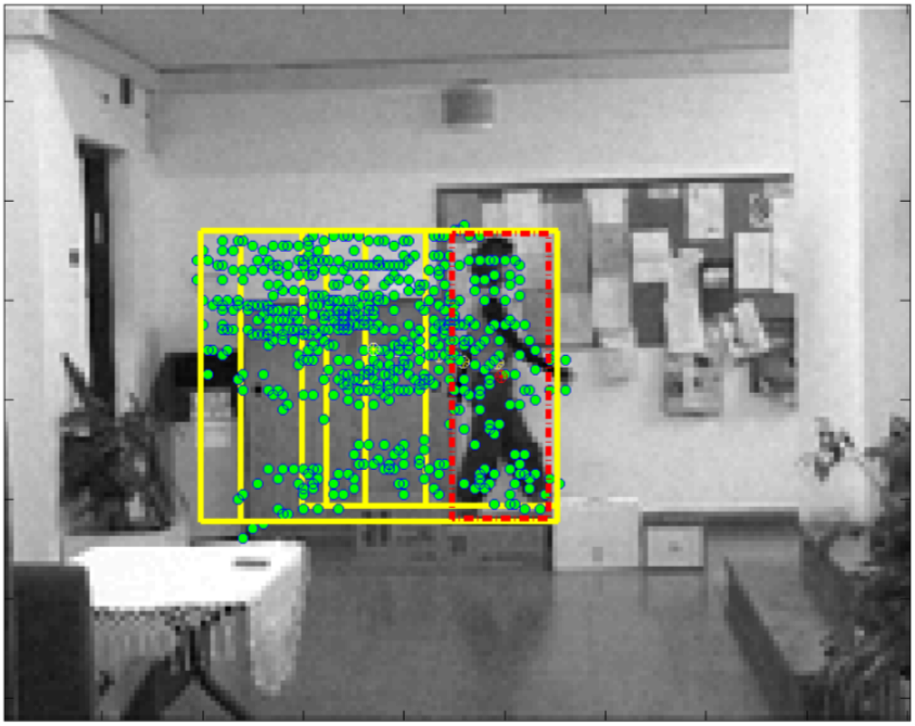

Figure 2.2.25: Cloud of interest points extracted in [25]

For example, Bregonzio et al. [25] proposed to use spatial and temporal locations of interest points for action classification. A cloud of interest points in spatiotemporal space is calculated and used for activity recognition (Figure 2.2.25). A bank of Gabor filters with different orientations is employed to extract the interest points. The authors claimed that detected interest points with this method are more related to human movement than those proposed in [19]. The authors also defined a descriptor to describe the cloud of interest points. Their descriptor involves the shape, speed and density of clouds on one side, and the width/height ratio and speed of foreground object for each frame on the other hand. By concatenating these descriptors for all frames, a high dimension descriptor for an activity is created. The authors showed that their method is comparable to most of the methods in the literature.

Sparse High-Level Features

High-level sparse features are the last group of sparse features. These features are usually extracted from high-level representations of the human body. For example, these features may be extracted directly from the human skeleton.

Li et al. [44] proposed to use high-level information for scene classification. The authors mentioned that despite NLP applications, image processing paradigm used low-level or mid-level information. They have proposed to use "*Object Bank*" for scene categorization.

Inspired by the success of object bank, Sadanand and Corso [45] proposed to use action detection for action recognition, they called their method "*Action Bank*". They have used action spotting method, originally proposed in [46], to calculate a correlation volume for each action, then by using max-pooling they have built a descriptor for each volume. A standard SVM classifier used for classification of these sparse features and it produced good results for more realistic datasets.

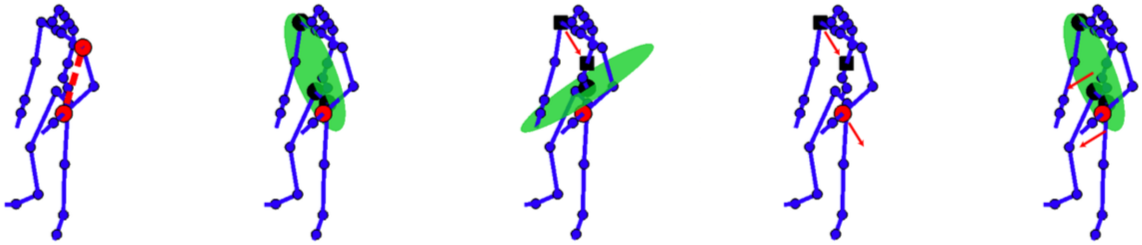

Figure 2.2.26: Feature descriptors extracted from skeleton [26]

Even though the appearance based methods, notably sparse feature descriptors, are simple methods and perform relatively good in complex situations, humans are questionable to use this kind of clues for activity recognition. Yao et al. [26] showed that even in the presence of noise, most of the features extracted from articulated human poses could outperform appearance-based methods using the same learning algorithm. The authors propose that using a combination of appearance-based and pose information may lead to a better classifier. As a result, a descriptor defined based on the joints locations, their speeds and their distances to the body plane [1].

In a similar experiment, Jhuang et al. [47] mentioned that despite the success of low-level features on easy datasets, their performance on more challenging datasets, are not satisfying. To find the best method for activity recognition the authors compared low-level, mid-level and high-level features for activity recognition. They first selected images that contain 21 single human activities, then annotated them with ground truth information of the 2D joint position, scale, viewpoint, segmentation, puppet mask, and puppet flow. The extracted descriptors from high-level pose feature outperformed visual-based descriptors. They have also shown that context information will not improve the results if the pose features are correctly extracted. By adding some noise to the ground truth joint position, they proved that their proposed pose features are robust to errors in the estimation of joint positions.

In another work, Pishchulin et al. [48] applied dense trajectories [21, 22] and human pose information [49] to the challenging dataset of "MPI Human pose." They showed that dense trajectories are more suitable when the number of classes is high (namely 491 activity class). They mentioned that pose based methods and dense trajectory methods are extracting different types of information, so by combining this information, higher performances could be achieved. Besides, they have claimed that extracted information from the background (context) can help the classifier to achieve higher performance. They discovered that dense trajectories performance increases in the presence of multiple persons.

2.3 Trajectories as mid-level sparse features

Trajectories have been used in the literature for human activity recognition. This section focuses more on the research methods that benefitted from trajectories for human activity recognition.

[1] The body plane determined by three other joint locations.

Matikainen et al. [50] used KLT tracker to track a fixed number of feature points in time. They defined a trajectory snippet as $S = \{P_t - P_{t-1}, P_{t-1} - P_{t-2}, ..., P_{t-l+1} - P_{t-1}\}$. This snippet merely captures the displacements that happen for each frame. Then, k-Means with a standard Euclidean distance metric is used to cluster these trajectories in different clusters. They call each of this clusters a trajecton. In this sense, each trajecton is the quantized version of a trajectory and represents a specific category of trajectory. Later, these trajectons are used in a standard BOW method for classification of different activities. They have also argued that trajectories that are representing similar motions in spatially close areas should belong to the same body moving in front of the camera. The authors proposed a method to find the center of this cluster and add affine transformation information to trajectories to represent the motion of various parts of the body. Finally, the standard BOW method and SVM are used for clustering.

In a similar work, Messing [51] used KLT to track key points of a video. They argued that KLT has fast, and even real-time performance implementation over GPU. The velocity of each feature's point over time, is calculated and quantized into "velocity history". They also augmented their trajectories with additional information. They augmented the original trajectory descriptor with information about the absolute position, appearance, and color to improve its classification power. They created a generative model on their augmented trajectories.

Wang et al. exploit trajectories in separate contributions [21, 22, 52]. By using the Farnback optical flow, the optical flow field of video is calculated. This step is the most time-consuming step. A dense grid on top of each frame is used to dense sample the field of view. In the ideal situation, all of these points should be tracked in time, but some points lack enough texture and therefore are not easy to track. These points, have a small self-similarity measure. The points that the smaller eigenvalue of their autocorrelation matrix is less than a threshold were removed from tracking procedure. They have also removes points with sudden movement or stationary trajectories as these points are erroneous and noninformative. They have also defined a trajectory shape descriptor to encode local motion patterns. Having a trajectory of length l, specified by a sequence of points $T = (P_t, P_{t+1}, P_{t+2}, ..., P_{t+l})$. The shape descriptor proposed by Wang et al. [52] defined as:

$$S' = \frac{(\Delta P_t, ..., \Delta P_{t+l-1})}{\Sigma_{j=t}^{t+l-1} ||\Delta P_j||} \tag{2.3.1}$$

In which $\Delta P_t = (P_{t+1} - P_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$. Equation 2.3.1 captures the normalized displacements over time. They have also defined trajectory aligned descriptors, which are merely traditional HOG, HOF and MBH descriptors calculated on the aligned neighborhood of each trajectory.

To improve the dense trajectories robustness, Wang et al. [22] proposed to model the camera motion between each consecutive frames. They assumed that there is a homography that can describe the motion between two frames. They argue that this assumption holds when the camera motion is not significant between two frames. The independent moving objects in the scene, should not affect this homography. They employed and matched SURF features, which are robust to motion blur, to find the correspondence between any two consecutive frames. To estimate the homography robustly, they have employed the RANSAC method. This homography then used to cancel out the camera motion before calculating the trajectories and other descriptors.

Sun et al. [53] proposed to track scale-invariant feature transform (SIFT) feature points. They have suggested using SIFT feature descriptors to match each frame feature point to the next one. They captured the point-level context by averaging SIFT descriptors over the course of each trajectory. To capture the dynamics of motion, they used an HMM to model displacements of a trajectory over time. They have extracted features at different levels and used multichannel nonlinear SVM for human activity recognition.

Deep learning methods have gained more attention recently, and they can perform very good in many different tasks. However, they need a large and correctly labelled dataset for the task in question. One of the best methods in this area is the ConvNet [54] that models the temporal and spatial space in two separate neural networks. The ConvNet can produce comparable results to dense trajectories [21]. The amount of data that deep models need to train can explain the reason why ConvNet could not outperform dense trajectories. The deep models usually need huge datasets, while most of the existing datasets for human activity recognition is small, compared to the number of different ways that humans can perform an activity. Another important thing is that the ConvNet treat the spatial and temporal channels separately without trying to find a connection between them.

A more successful approach for deep learning based methods is trajectory-pooled deep-convolutional descriptor (TDD) [55]. Although it is not directly relevant to trajectory shape descriptor, this method, proposed to use trajectories introduced in [22], to align neighborhood before calculating the ConvNet descriptors. They demonstrated that TDD descriptors could achieve the state of the art performance.

2.4 Learning from a set of sparse features

Sparse representation methods, represent a video by a set of independent features. Formally, a video can be represented by a set of feature descriptors as:

$$S = \{D_k | D_k \in \mathbb{R}^N\} \tag{2.4.1}$$

where N is the dimension of the local descriptors.

Existing machine learning methods in general and SVM in particular, expect data as a vector of predetermined size. As a result, each set of these features should be represented by a vector. Different methods have been proposed in the literature. Here we explain two favorite techniques in the literature.

2.4.1 Bag of Words

One of the conventional methods to convert sparse sets to a vector is based on the bag of words (BOW). It is inspired from the text processing paradigm, where a collection of significant words are extracted from the text and are placed in different bags [56]. The descriptor for a bag is created based on the number of times each word appears in the bag.

Since vectors representing local features are continuous, K-Means or a similar algorithm can be used to quantize the vectors. First, K-means is trained by a sample of all videos. Then, this model is used to cluster all the feature vectors of each set. The number of items in each cluster is used to create a descriptor.

More formally, K-Means is used to cluster local descriptors into N clusters, where N is called the number of words. For each video j a descriptor C_j is defined as:

$$C_j \equiv (c_{j1}, c_{j2}, ..., c_{jN}), c_{ji} \in \mathbb{N}_0 \tag{2.4.2}$$

where, c_{ji} is the number of trajectories of video j that are clustered in cluster i.

These values are then normalized based on their min and max values from all learning videos.

$$n_{ji} \equiv \frac{c_{ji} - \min_{k \in L} c_{ki}}{\max_{k \in L} c_{ki} - \min_{k \in L} c_{ki}}, n_{ji} \in [0, 1] \tag{2.4.3}$$

where L represents the set of learning video indices.

Finally, each video j is represented by a vector V_j:

$$V_j \equiv (n_{j1}, n_{j2}, ..., n_{jN}) \tag{2.4.4}$$

2.4.2 Fisher Vector Encoding

Fisher Vector Encoding (FVE) is another method that produced good results. In FVE, the generative and discriminative mathods are combined [57] and, the first and second order statistics have been used for encoding [22] (in contrast to first order statistics of BOW).

Instead of using K-Means for clustering, Expectation Maximization (EM) is used to cluster data into K Gaussian Mixtures. The created Gaussian mixture model (GMM) is used to estimate the means, variances and prior probabilities of the mixtures. Let $\theta_k = \{\mu_k, \sigma_k, \omega_k\}$ represent the parameters for component k. Then, the posterior probability of observations x_i with respect to to k, q_{ik}, is calculated as follows:

$$q_{ik} = \frac{exp[-\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)]}{\sum_{t=1}^{k} exp[-\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)} \tag{2.4.5}$$

This information is used to calculate the mean deviation u_k and covariance deviative v_k.

$$u_{jk} = \frac{1}{N\sqrt{\omega_k}} \sum_{i=1}^{N} q_{ik}(\frac{x_{ji} - \mu_{jk}}{\Sigma_{jk}}) \tag{2.4.6}$$

$$v_{jk} = \frac{1}{N\sqrt{2\omega_k}} \sum_{i=1}^{N} q_{ik}[(\frac{x_{ji} - \mu_{jk}}{\Sigma_{jk}})^2 - 1] \tag{2.4.7}$$

where j spans over the vector dimension and k spans over different components of the mixture model.

The size of u_k (or v_k) is $K \times D$, which only depends on the number of Gaussians (K) and dimension in which data is represented (D) (not the number of samples N). The final descriptor is created by concatenating these two vectors, and thus has a dimension of 2DK. Usually, further improvement is achieved by l_2 normalization and the use of a nonlinear additive kernel.

2.5 Conclusion

Numerous methods have been proposed for human activity recognition in the literature. The most popular methods are based on sparse representation, which is robust to most of vision based difficulties. These Methods can work in more realistic situation, where the camera is not stationary.

Low-level features are trendy and well studied in the literature. In some experiments, these features have benefited from background information. The strength of these methods come from ignoring spatial and temporal information. There is another face to this ignorance. Since activity is represented as a set of isolated features, sparse methods are unable to model long or complicated activities. These methods have another limitation. When the number of action classes increases, the low-level representation performance decreases fast.

Instead of throwing away useful structural information, mid-level and high-level representations can take advantage of them. For example, in sparse high-level representation, some authors used human skeleton angles or positions for activity recognition. This way sparse high-level representations are more discriminative than low-level sparse representations. The cost of this improvement is the use of depth camera or mocap sensors.

CHAPTER 3

The Bag of Micro-Movements for Human Activity Recognition

3.1 Introduction

This chapter covers the first experiment with disparities in human activity recognition. It proposes a new method for human activity recognition based on the bag-of-words (BOW) method [58], which is inspired by the success of BOW assumption in document classification problem. BOW states that the topic (or class) of a document can be determined solely by looking at the words that appeared in that document[1], regardless of their place of appearance. In vision community, sparse features from input videos are extracted and treated as words in a text. Since the obtained features usually are not discrete, some quantization (clustering) method is used to assign the continuous feature descriptors to the discrete words (clusters). The bag of words algorithm tries to match two documents based on the words that appear in both documents. In vision community, this translates into matching two videos based on the same words (similar neighborhoods) appearing in two videos. BOW is a powerful method because it removes complexities related to the duration of an activity or the speed of it.

Different feature point extraction methods have been proposed in the literature [17, 18, 19, 21, 22], each of them have their pros and cons, but almost all of them follow the same approach. First, the points of interest (usually corners) are extracted [17, 18, 19, 20, 58]. Then, a descriptor for each interest point is calculated by looking into its

[1] More accurately by the number of times a word appeared in a document compared to the number of times it appeared in other documents

neighbourhood [17, 19, 43]. These descriptors are calculated directly or indirectly from pixel values around a feature point. Here we call them appearance based feature points because these descriptors encode the appearance of their neighborhood. Even though BOW showed promising results, the type of features that was used is unlikely to be used by humans for activity recognition.

This chapter proposes a new descriptor, which encodes the motion information in a way that could be used efficiently by BOW algorithms. Our work is inspired by [52] which have introduced the motion trajectory descriptors. Our contribution is to combine disparity maps with motion information to improve the motion descriptors that we refer to them as micro-movement descriptors. Disparity maps, used in this chapter, are solely the Euclidean distance between similar points in left and right images, which are very easy to extract, and they do not need camera calibration, while they still provide a depth clue. We believe that disparity maps have enough discriminative information for many applications including HAR. The result of this research published in [59]. A more accurate method for disparity calculation with image rectification proposed in Chapter 6.

3.2 Motivations

The motion is a good clue for activity recognition and many different methods used motion information for human activity recognition with different approaches [4, 5, 7, 8, 11, 12, 13, 14, 35]. Some methods have tracked the location of human joints in 3D space[4, 5]. Other methods tried to track joint locations in 2D image plane[7, 13]. These methods usually rely on human skeleton extraction, but skeleton extraction directly from 2D images is still prone to errors. Some other methods have used only parts of the body for activity recognition. For example, [8] tracked the hand positions of a human and used these trajectories for activity recognition. This method is limited to activities that could be done only by hands. Some methods [14, 35] extracted 3D objects in spatiotemporal space. Other techniques used the compression of the motion into rigid 2D images [11, 12].

One of the most successful methods for activity recognition, which showed promising results, is the bag of words (BOW) [58]. This method is based on the sparse representation of activities; i.e., each activity video is represented by a set of isolated feature descriptors. Traditionally, these feature descriptors were directly or indirectly calculated based on the appearance of the neighborhoods around the feature points [17, 18, 19]. Their 2D image counterparts inspired most of these descriptors. That explains why most of the sparse feature extractors used appearance based information and simply neglected the motion. One exception is the method proposed in [52]in which they used dense trajectories and defined a trajectory descriptor. They have also used these trajectories to align feature point neighborhood frames and made a traditional feature point over the aligned neighborhood. In their experiment, trajectory aligned vision-based feature points showed better performance and, later [21, 22] they only used motion to align the feature point neighborhood frames.

It has been suggested in [52] that tracking interest points in a 2D image using KLT tracker yields high-grade results. The captured movements happen in the x-y image plane and can be expressed in the number of pixels. Each image interest point (x, y) represents a space point (X, Y, Z) and the relationship can be expressed by the distance of the space point to the camera and intrinsic and extrinsic parameters of the camera. So the captured movement depends on the depth of scene points changes, which is unknown. We believe that combining depth information with 2D trajectory information can improve the results.

One solution is to capture the depth with the help of active cameras. Although off the shelf active cameras are useful, their functionality is limited. Since they are using time of flight (TOF) calculation to estimate the depth, they can work in low to moderate resolution, and they can cover a specific range (between one and three meters depth). Besides, existing active cameras are limited to indoor environments.

Another solution is to use stereo cameras and triangulation. Let us assume a space point P is mapped to P_1 in the first image and P_2 in the second image. It is possible to calculate the coordinates of P in the scene coordinate system by having the coordinates of P_1 and P_2 in the image plane coordinate systems. Furthermore, let us assume that the two cameras have the same orientation and the images are horizontally aligned and coplanar. In this situation, it can be shown that the depth of point P depends only on the baseline[2] and the focal

[2] The distance between two centers of projections

length[3] of two cameras. In practice, one should have the camera calibration information, including effective focal length and lens distortion parameters. Such configuration is hard or impossible to achieve in some applications.

Even though the 3D information is beneficial in transforming motion in image coordinate system to motion in scene coordinate system, this information is hard to extract. Furthermore, disparities are much easier to obtain and there is no need for camera calibration. In this chapter, we proposed to use disparity instead of 3D information to represent motion in a new coordinate system, which is similar to scene coordinate system.

3.3 Micro-movement descriptors

Our method captures the motions of interest points as the main clue. First the interest points from both left and right frames are extracted, and for each of them a descriptor is calculated. Here we have used opencv implementations of FAST corner detector [61] for interest point detection and SIFT descriptors [43] for feature descriptor calculation. Then this descriptors are used to match feature points between the left and right frames. Having the point correspondences, this descriptor is no longer needed. Each interest point is now represented as $I\left(x_{li}, y_{li}, d_i\right)$ in which $P_l\left(x_{li}, y_{li}\right)$ represents pixel coordinate of interest point i in left image and d_i is the distance between left and right frame calculated as an Euclidean distance:

$$d_i = \sqrt{\left(x_{ri} - x_{li}\right)^2 + \left(y_{ri} - y_{li}\right)^2} \qquad (3.3.1)$$

If we assume the cameras are aligned such that there is no y-displacement, i.e., $y_{ri} - y_{li} = 0$, then the above distance will be reduced to $d_i = |x_{ri} - x_{li}|$ which is x-disparity. Since we are trying to reduce any precondition over camera placements, we have used the 2D Euclidean distance.

We have used KLT tracker to track the interest points in the left and right frames. We tracked the interest points for l consecutive frames before recalculating the interest points for the $l + 1$ frame. This way several trajectories of length l have been created.

After extracting trajectories, the displacement vector calculated based on the amount of

[3] The distance between the center of projection and the image plane

movement that each point has undergone. For example, if $l = 3$ and a sample trajectory T_i given by:

$$T_i = ([x_1, y_1, d_1], [x_2, y_2, d_2], [x_3, y_3, d_3]) \tag{3.3.2}$$

Then the displacement D_i is calculated as:

$$D_i = ([x_2 - x_1, y_2 - y_1, d_2 - d_1], [x_3 - x_2, y_3 - y_2, d_3 - d_2]) \tag{3.3.3}$$

Note that each displacement calculated between two consecutive frames (not left and right frames). The displacement contains the motion information that existed in the video. In general case:

$$D_{ij} = I_{i(j+1)} - I_{ij} \tag{3.3.4}$$

Where D_{ij} represents the component j of trajectory i and I_{ik} represents the interest point triplets (x_{ik}, y_{ik}, d_{ik}) in trajectory i. Note that when the length of T_i is l then the length of D_i would be $l - 1$.

We have defined an energy measurement for each trajectory, given by:

$$e_i = \Sigma_{k=1}^{l-1} |D_{ik}|^2 \text{ Where } |D_{ik}| = \sqrt{x_{ik}^2 + y_{ik}^2 + d_{ik}^2} \tag{3.3.5}$$

The energy of a displacement determines the amount of movement of the corresponding trajectory. Low energy trajectories will represent steady feature points in a video. These points are usually background points or points on the human body which are not moving in l consecutive frames. These points have no discriminative information; therefore they are removed by simple thresholding. This eliminates the trajectories with very low information.

The remaining of the displacements are mapped onto a three-dimensional space $(M(X, Y, Z))$ which has the characteristics of scene coordinates. From the stereo camera model and triangulation, it can be deduced that:

$$Z = fB/d \propto 1/d \tag{3.3.6}$$

$$X = uZ/f \propto uZ \propto u/d \tag{3.3.7}$$

$$Y = vZ/f \propto vZ \propto v/d \tag{3.3.8}$$

In which f is focal length and B is the baseline distance. Assume $D\left(u, v, d\right)$ represents a point in displacement coordinate system measured in pixel values. We calculated our micro-movement descriptor by normalizing a displacement as follows:

$$M\left(X, Y, Z\right) = \left(u/d, v/d, 1/d\right) \tag{3.3.9}$$

Where $M\left(X, Y, Z\right)$ is represented in an independent coordinate system. Movements in this space are similar to movements in the scene coordinate system.

3.4 Experimental Result

To the best of our knowledge, there is no stereo vision dataset for human activity recognition. Hence, it is hard to compare our proposed method to other methods in the literature. To demonstrate the effectiveness and discriminative power of our proposed micro-movements representation, we have created our own stereo-dataset. The latter contains 12 different simple activities.

Each activity is done four times by two volunteer actors, a male, and a female. The videos are recorded with two off the shelf cameras attached to a rigid bar. The videos are captured and recorded in VGA quality. Some sample frames of the dataset are shown in Figure 3.4.1. Figure 3.4.2 shows the cameras used to capture the movements. Figure 3.4.3 demonstrates the feature points extracted from the corresponding left and right frames and the points matching result.

After extracting the micro-movement descriptors, we cluster them using the well known K-Means clustering algorithm. In particular, each cluster represents a word. For each instance of activity in our dataset, we have counted the number of times each word appears in it. Then, we have made a vector of length w words, where w represents the number of clusters. For this experiment, we fixed the length of trajectories to $l = 9$ frames and $w = 400$ as a rule of thumb. We have used *"Bayes Net"* for classification of activities based on the word count vector. We were able to correctly classify 73.47% of the activities

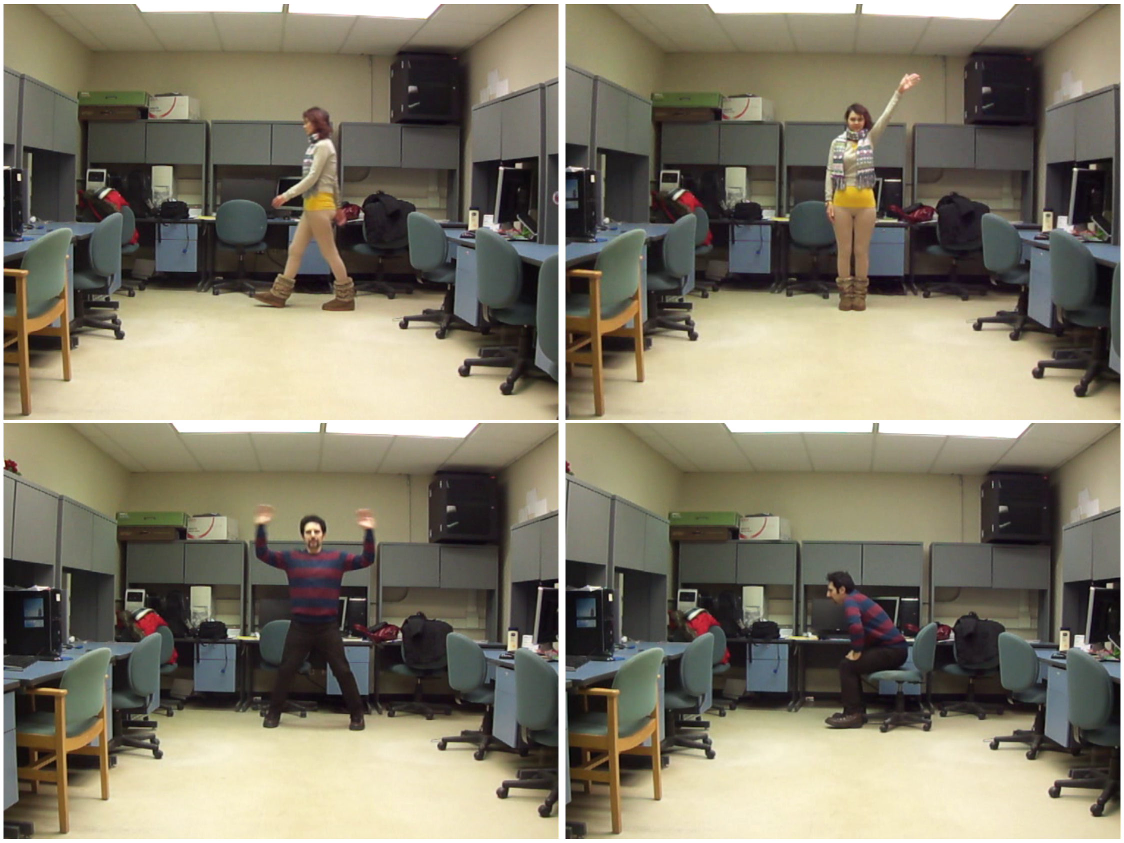

Figure 3.4.1: Sample frames from dataset demonstrating: walking left, hand waving, simple exercise and sitting down

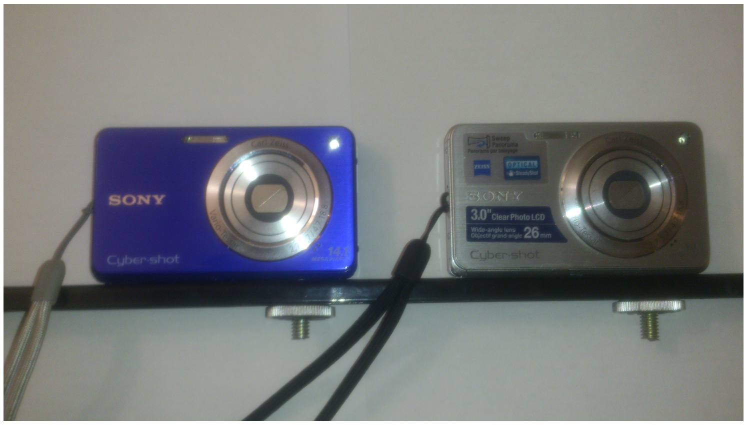

Figure 3.4.2: The stereo camera setup

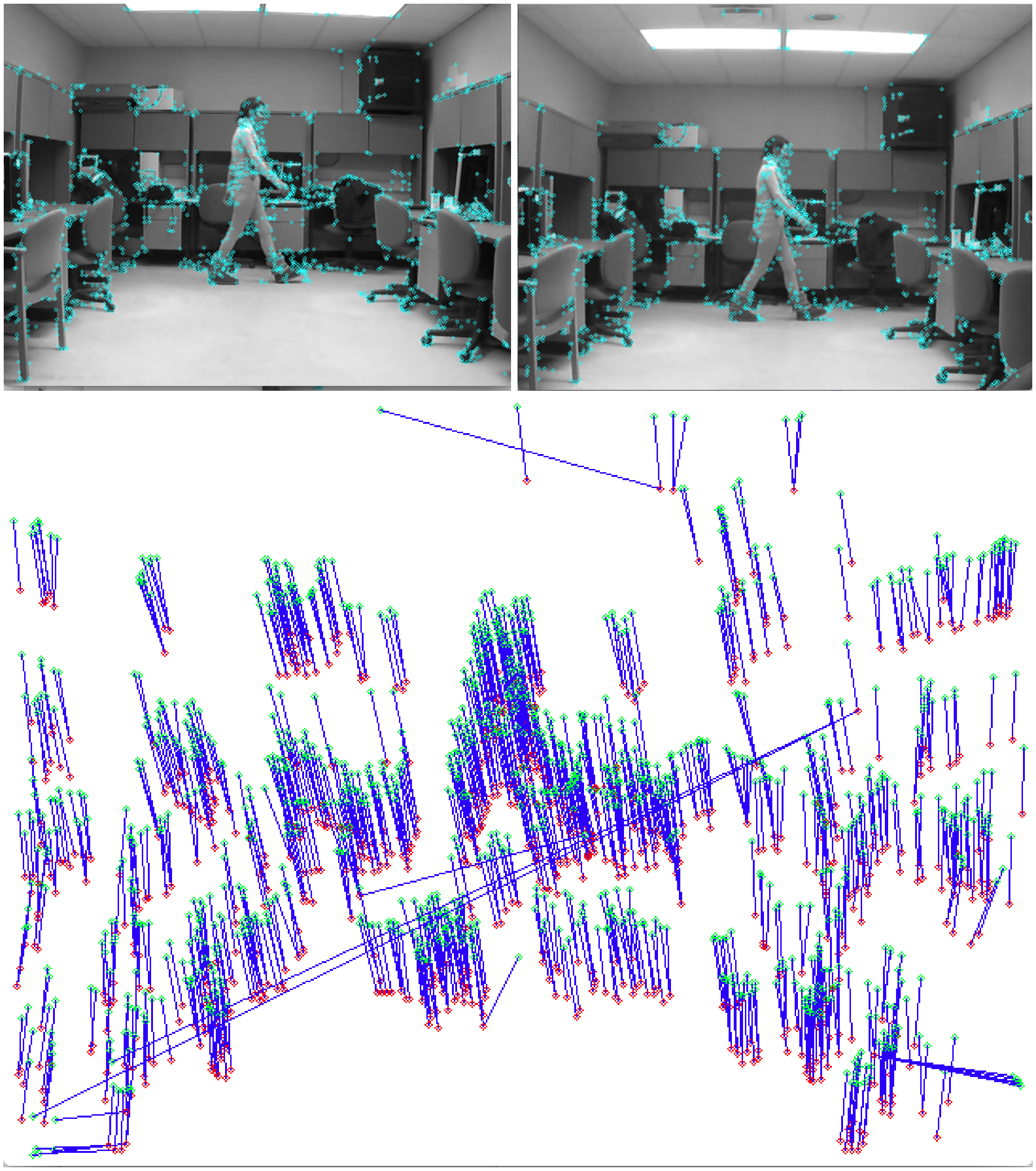

Figure 3.4.3: The left and right extracted feature points and their matching result on a sample frame

Table 3.4.1: The confusion matrix of twelve activities

Activity Name	Class	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**
Walking Right	**a**	10	0	0	0	0	1	0	0	0	0	0	1
Walking Left	**b**	0	11	0	0	0	0	0	0	0	0	0	0
Walking Toward Camera	**c**	1	1	2	3	0	0	1	0	0	0	0	0
Walking Away Camera	**d**	0	0	2	8	0	0	0	0	0	0	0	0
Hand Waving Right	**e**	0	0	0	1	5	0	0	0	3	1	0	0
Hand Waving Left	**f**	0	0	0	0	1	4	0	1	1	0	0	0
Jumping	**g**	0	0	1	0	1	0	10	4	0	0	0	0
Sitting Down Front View	**h**	0	0	0	0	0	0	1	10	4	1	0	0
Standing Up Front View	**i**	0	0	0	0	0	0	0	2	11	0	1	0
Sitting Side View	**j**	0	0	0	0	0	0	0	1	1	13	0	0
Standing Side View	**k**	0	0	0	0	0	0	0	0	1	0	14	0
Jumping Jack	**l**	0	0	0	0	0	1	0	0	1	1	0	10

Table 3.4.2: Six activity confusion matrix

Activity Name	Class	**a**	**b**	**c**	**d**	**e**	**f**
Walking	**a**	36	1	3	0	0	1
Hand Waving	**b**	1	11	0	4	1	0
Jumping	**c**	1	2	13	0	0	0
Sitting Down	**d**	0	0	2	27	2	0
Standing Up	**e**	0	0	0	4	25	0
Jumping Jack	**f**	0	1	0	1	0	10

without any parameter tuning. It is hard to compare this value with other works. The nearest work to ours is the image plane motion descriptors of [52]. They achieved 67.2% accuracy on YouTube dataset. Their trajectory aligned descriptor hit 83.9% which is the state of the art.

The confusion matrix of twelve activities are represented in Table 3.4.1. We should emphasize that our result is preliminary and we improve them in the next chapters. The main goal here is to demonstrate that trajectories have discriminative information.

Our test setting neither designed nor optimized for online processing, however with current setting the extraction of features is done in 8.3 frames per second on a single thread ran on a 2.8 GHz Core i7 CPU. With some improvements, one might be able to implement it in real time, but the original BOF algorithm should also be altered to work in an online manner.

To further demonstrate the flexibility of our classifier, considering that some of the activities in our dataset are very similar and they typically have the same name in our natural language, we have combined the similar classes to examine the discriminative power of our descriptors. We summarized our activities into six different classes. Using same Bayes Net classification method without parameter tuning, we have achieved 83.56% accuracy. The confusion matrix of this experiment is shown in Table 3.4.2.

3.5 Conclusion

This chapter proposed and implemented the bare idea of using disparity information and fusing it with trajectories. Although the proposed descriptor is easy to extract and can discriminate between different activities, its performance needs to be improved. This descriptor is not bound to human activity recognition task. It is useful for any other video analysis problem, where the movement is the discriminative clue. By using disparity information, we are taking advantage of 3D structural data, while eliminating the requirement for calibrating the cameras. A big dataset should be created to assess the effectiveness of the proposed method, properly. Chapter 6 covers the proposed bigger dataset.

CHAPTER 4

Trajectories for Human Activity Recognition

4.1 Introduction

The previous chapter showed that adding disparity information to trajectories can improve the trajectory discriminant power. This chapter goes into different trajectory shape extraction methods and compares them with each other. These experiments aim to discover which approach for 2D trajectory extraction is better in specific scenarios and why.

As we discussed earlier, the sparse representation methods were more popular in the past decade. Sparse feature vectors can represent low-level, mid-level or high-level information. There is a handful of research done on low-level features [16, 17, 18, 19, 23, 42, 52, 58]. Although the low-level features can produce relatively good results, they usually use the appearance-based information in small neighborhoods, which makes them dataset dependent. Besides, high-level features, like the human skeleton, have shown promising results [37, 62]. However, they are not easy to extract from video and, the best existing methods are still prone to error [26, 45, 63]. Moreover, mid-level features can be easily extracted with good confidence, and they represent higher level information than their low-level counterparts.

Trajectories, as mid-level sparse features, are the flow of 2D interest point locations in time. They are relatively easy to extract, and they capture the motion shape (Section 4.2). Trajectories have been proven useful to align small neighborhood frames before calculating traditional descriptors like Histograms of Oriented Gradients (HOG), Histograms of Optical Flow (HOF), or Motion Boundary Histograms (MBH) [52]. Trajectory shape descriptors are also shown to have useful information for human activity recognition.

There are at least two reasons to believe that trajectory-based shape descriptors can still be used for human activity recognition.

1. It has been shown in [2] that human subjects can guess the type of activity performed by another human subject, only by watching a video of moving points attached to the actor. Hence it is safe to assume that trajectories carry the discriminant information.

2. Traditional sparse feature descriptors, like HOG, HOF, and MBH, encode low-level information that is unlikely to be used by humans for activity recognition. On the other hand, trajectories can be mapped to existing motions in a video, making them mid-level features.

This chapter emphasizes the importance of trajectories for human activity recognition and introduces three different trajectory extraction methods for HAR. In particular, it compares these methods with each other and with the trajectory-based method proposed in [52]. It also lists the pros and cons of using trajectories for human activity recognition. Moreover, it investigates the effect of trajectory length on the classification accuracy (Sections 4.4.1, 4.4.3). The comparisons demonstrates that sparse sampling can produce comparable results to dense sampling, with the advantage of using fewer data.

4.2 Proposed Trajectories for Human Activity Recognition

Trajectories are trails of 2D spatial feature points in time (Figure 4.2.1). 2D feature points could be extracted by any feature detector algorithm. Formally, a trajectory T_k is an ordered list of spatial locations, in $l + 1$ consecutive frames, where l is called the length of the trajectory.

$$T_k \equiv (p_0, p_1, p_2, ..., p_l), p_i \in R^2, i = 0..l \tag{4.2.1}$$

There are different methods for extracting trajectories from videos. Here we used three extraction algorithms to produce different trajectories, as shown in Figure 4.2.2. We have

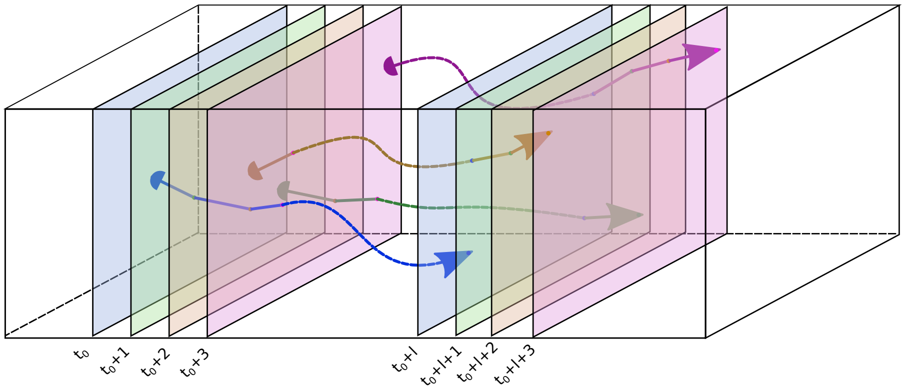

Figure 4.2.1: Tracking of feature points for l consecutive frames in spatiotemporal space (best seen in color)

examined these trajectories and have explained the pros and cons of each of them in Section 4.4.3.

4.2.1 Interest Point Tracking

The first algorithm is the tracking of 2D Interest Points based on their appearance; we call it *"IP algorithm"*. There are two assumptions for this algorithm: (1) interest points remain similar in appearance and, (2) their positions do not change a lot between consecutive frames.

We have first extracted interest points from each frame of the video, using *FAST* method [61]. Then, *SIFT* descriptor is calculated for each of these points [43]. Starting from the beginning of the video, for all interest points of the current frame, the best matches in the next frame are found, based on spatial distance and the points' descriptors. Once matched, they are to form trajectories. Interest points in the following frame that did not match any point from the current frame are considered starting points of new trajectories. When a trajectory length reaches l, we consider it as a full trajectory and the endpoint will not be matched against next frame feature points.

4.2.2 Lucas-Kanade Trajectories

This method is based on Lucas-Kanade (LK) optical flow algorithm [64]. First, interest points in each frame are extracted, using *FAST* method [61]. For each interest point in

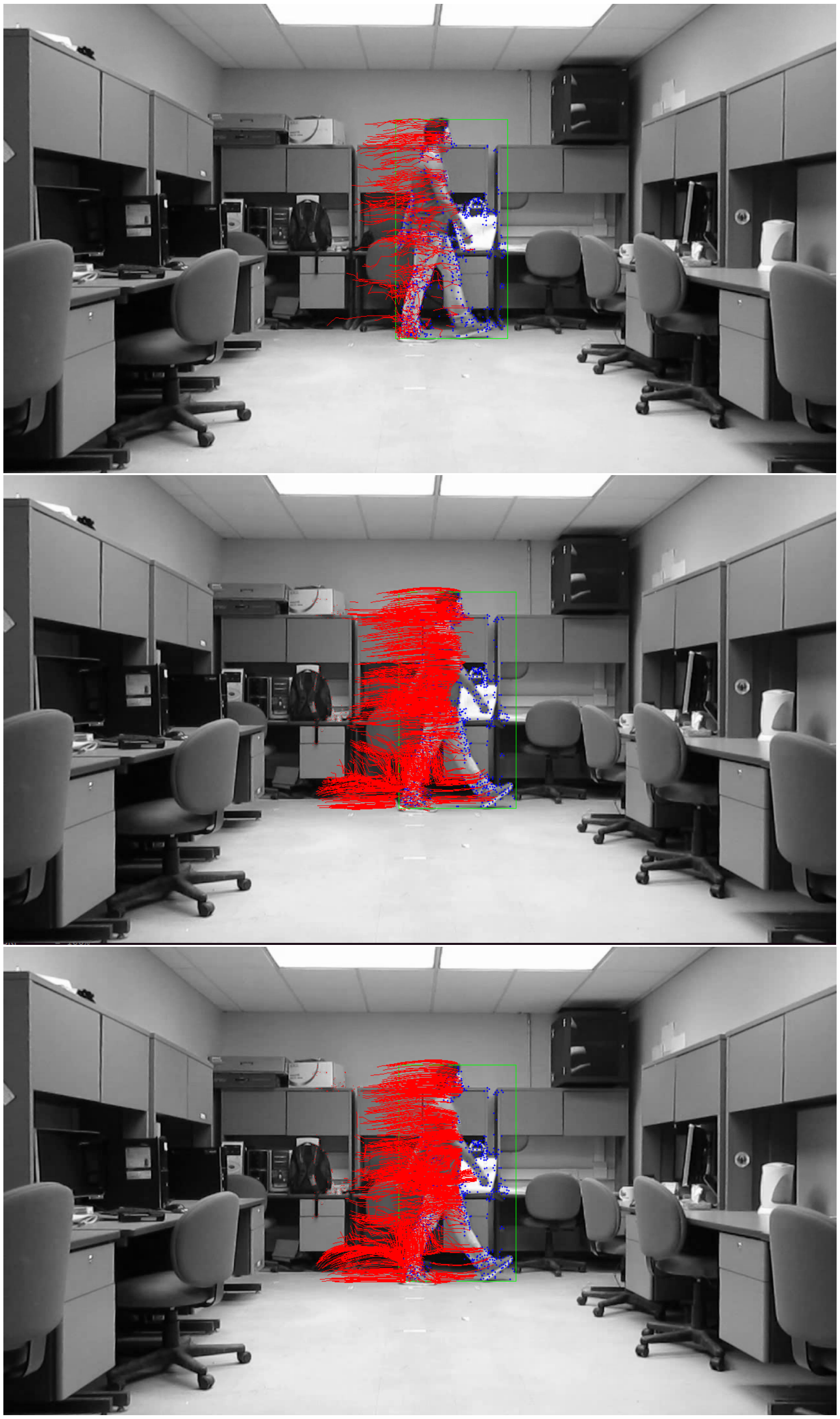

Figure 4.2.2: From top to bottom: sample output of IP, LK, and FB trajectories extracted for walking from left to right (best seen in color)

the current frame, the Lucas-Kanade feature point tracking algorithm is used to find the most probable location of this point in the next frame. These two points are connected as being part of a trajectory. In the next frame, these locations and other new feature point locations will be mapped to their next frame locations as well. The process continues until the trajectory reaches length l, when it is considered a full trajectory.

4.2.3 Farnback Trajectories

The third algorithm used here is based on the Farnback (FB) optical flow algorithm. Farnback is a newer optical flow algorithm compared to Lucas-Kanade. The main advantage of the Farnback method is that it calculates a dense optical flow field.

The algorithm is quite similar to LK trajectory extraction algorithm (Section 4.2.2). First, interest points of each frame are extracted. Then the optical flow field is calculated for the video. The location of current frame interest points in the next frame is found based on the optical flow field. These points are connected as being part of a trajectory. The process continues similar to LK algorithm until trajectories reach length l.

4.3 Trajectory Shape Descriptor

After extracting the trajectory, its shape should be encoded in a vector. Similar to [52], we have defined the trajectory shape descriptor as the normalized derivative of the trajectory.

$$D_k \equiv \frac{(d_1, d_2, ..., d_l)}{\Sigma_i \|d_i\|}, d_i \equiv p_i - p_{i-1}, i = 1..l \tag{4.3.1}$$

where D_k is a simple shape descriptor for trajectory T_k, $\{p_i\}$ belong to T_k (from Equation 5.3.1) and $\|.\|$ denotes the $L^2\ norm$. We have used this descriptor to cluster trajectories into different categories. Despite its simple nature, the trajectory shape descriptors are very efficient. They capture the way the trajectory behave while ignoring the spatial place of it.

This way, each video will be represented as a set of trajectory shape vectors. The cardinality of these sets are not the same, and it makes the video matching hard. One of the well-known approaches proposed in the literature to cope with the similar situation is the

Bag of Word method.

4.4 Experiments

4.4.1 Setup

To calculate the accuracy, we have used the leave one out method, which is similar to N-Fold cross-validation. All instances of one actor are taken out from the dataset, and the rest are used for learning. At the test stage, only those specific samples are classified. The process is repeated for all actors, separately, and the overall classification accuracy is calculated by considering all classifications.

For the learning part, the standard BOW algorithm is used, and as suggested in [52], the number of words is set equal to 4000 and, only 100K samples are used to learn the KMeans models. For classification, libSVM [65] with RBF Kernel was used, and the C-SVC cost set to 1500.

All of the related code has been implemented in C++ using OpenCV. The tests were run on a 3.7 GHz 8 core UNIX-based computer.

4.4.2 Dataset Used

Originally we are working on stereo vision, and there are only a few specific datasets available for this reason, so we have created a stereo dataset for human activity recognition that involves 11 actors performing many different activities. The dataset has been recorded in real life everyday office setting with complex background. Each actor repeated each activity at least five times. The dataset recorded by two off-the-shelf similar video cameras pointed at the subject from the same direction with a slight angle. Chapter 6 provided the full description of the dataset.

For the tests performed in this chapter, we only need videos from one camera. We have selected ten activities of three different actors from left camera. The selected activities are hand clapping, jumping, skipping, picking up, pushing, running, sitting down, standing up, walking and jumping jack. Whenever the selected activities contained movement (such

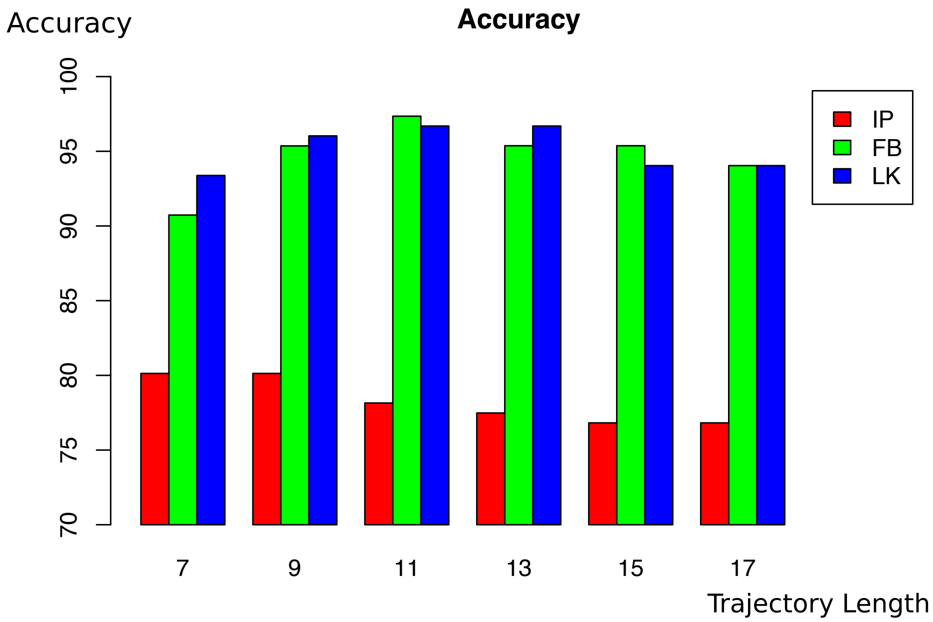

Figure 4.4.1: Comparison of different algorithms with different length, note that the chart is cut for better visibility

as walking, running, or jumping), all the instances were selected in a way that all the movement happens from left to right (e.g., walking from left to right).

4.4.3 Obtained results

Figure 4.4.1 and Table 4.4.1 summarize the results of our different tests. As it can be seen in Table 4.4.1, Lucas-Kanade (LK) and Farnback (FB) trajectory extraction algorithms are producing competitive results, while interest point tracking (IP) on the other hand is not as good. The best result obtained was 97.03% by FB, with length 11. The LK algorithm produces comparable results with a 96.69% recognition rate. The best result of the IP algorithm yield a modest 80.13% recognition rate. We believe that there are at least two reasons for IP's poor result. First, the number of trajectories extracted by the IP algorithm is quite low. Second and more importantly, the IP tracking algorithm is not producing good trajectories (from the visual point of view).

Table 4.4.1 has the dense trajectory results in addition to our proposed sparse algorithms. The dense trajectories are extracted using Wang implementation of dense trajectory extraction, then a similar approach (as explained in Section 4.4.1) was used to evaluate the

Table 4.4.1: Results obtained by different tracking algorithms and different trajectory length over recognition rate.

		Length of Trajectory					
Category	Algorithm	7	9	11	13	15	17
Sparse	IP	**80.13%**	**80.13%**	78.15%	77.48%	76.82%	76.82%
	FB	90.73%	95.36%	**97.35%**	95.37%	95.37%	94.04%
	LK	93.38%	96.03%	**96.69%**	**96.69%**	94.04%	94.04%
Dense	Trajectories	92.72%	92.72%	93.38%	94.04%	94.04%	95.36%

Table 4.4.2: Comparison of dense sampling and sparse trajectories

Category	Algorithm	Accuracy
Sparse	IP	80.13%
	FB	97.35%
	LK	96.69%
Dense	Trajectories	94.04%
	HOG	86.76%
	HOF	97.35%
	MBH	96.03%

accuracy.

One might expect that the dense trajectory results be similar to FB algorithm since the algorithms are very similar. However, Table 4.4.1 suggests that FB almost always produced slightly better results compared to dense trajectories. This can be explained by the fact that FB trajectory extraction algorithm employed here, used a simple corner detector instead of dense sampling and these points are more comfortable to track. All in all, dense sampling trajectories could reach 95.26% accuracy while FB sparse trajectories reached 97.35%.

Table 4.4.2 summarizes the traditional appearance-based descriptors. The length of trajectory is set to 15 (as suggested in [52]) for all dense results reported in Table 4.4.2. As it can be seen, the FB and LK trajectories produced comparable results to HOF and MBH algorithms, while HOG, which only uses appearance based information, could not exceed 87%. Recall that FB and LK are using sparse sampling, in contrast to dense sampling for HOG, HOF, and MBH.

Figure 4.4.2 shows the confusion matrix of the best obtained classification. As it can

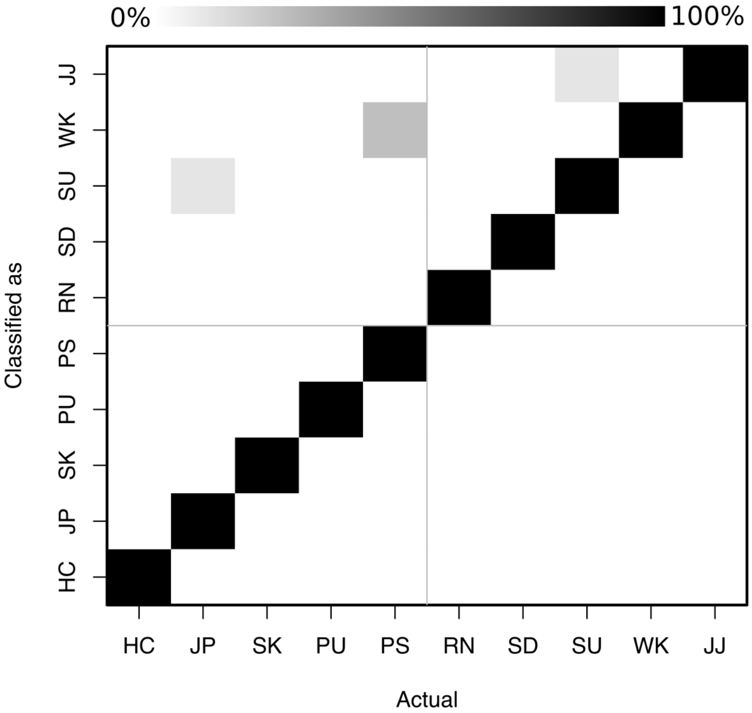

Figure 4.4.2: The sample confusion matrix

be seen, the most confused classes are the PS (Push heavy object on the floor from left to right) and WK (Walking from left to right), which are very similar. In particular, PS is essentially walking from left to right while pushing an object. The other error came from confusing JP (jumping), SU (Stand up) and JJ (Jumping Jack). Even though these classes share some level of similarity, they can be put in different classes. This reveals that in some situations, trajectories might not be discriminant enough. This is the case for activities that are similar form the motion point of view, for example, Jumping, jumping jack and standing up produce upward trajectories, are confusing at the classification stage.

4.5 Conclusion and Future work

This chapter compared three different trajectory-based algorithms for human activity recognition: the Interest point (IP), Lucas-Kanade (LK) and Farnback (FB) trajectories. These trajectories are easy to extract, and they capture the motion information of the video.

The tests done in this chapter, demonstrate that trajectories carry good discriminant information and therefore are useful for HAR. More specifically, FB and LK algorithm proposed here could reach 97.35% and 96.69% accuracy respectively while the Dense Trajectories proposed by Wang only reached 95.36%. It is also worth mentioning that FB and LK used corner detector on moving parts of the video instead of processing the whole field of view, which makes this approach faster for real-life applications.

On the other hand, we have also found out that trajectories might not be the best choice for HAR. Some activities are similar from motion point of view; these activities yield similar trajectories, and hence trajectories are not useful in this situations.

Enriching trajectories by adding another kind of discriminant information might lead to better classification performance. Moreover, the shape descriptor we have used here is a simple one and defining a better shape descriptor might also improve the overall classification performance. Next chapter, investigate this further.

CHAPTER 5

Improving Trajectory Shape Encoding

5.1 Introduction

The previous chapter demonstrated the effectiveness of trajectories for human activity recognition. On the downside, the naive trajectory shape encoding algorithm used in that chapter is not able to capture all the discriminant information the trajectories have. This chapter investigates this problem further and proposes a better trajectory shape descriptor, which is a superset of existing trajectory shape encoding algorithms.

As it has already mentioned, sparse representation methods describe a given video by a set of sparsely sampled features, regardless of their location (spatial and temporal). Then, this feature set is mapped to a fixed-sized vector so that it can be used by any existing machine learning method. Many sparse representation methods proposed in the literature rely on local appearance information. Few other methods have used the motion of feature points for this matter (see Section 5.2).

The previous chapter defined trajectories as the trails of interest point over time. Trajectories are also useful for aligning small neighborhoods in consecutive frames for extraction of traditional local features.

Although the trajectory aligned descriptors can produce comparable, or slightly better results, they are more demanding regarding computations. Once the trajectories are obtained, one has to align the frames based on trajectories and then calculate traditional descriptors. There are many such trajectories in a video, which means a lot of processing time is needed. This chapter aims at overcoming this issue by finding a better way of using trajectory shape information directly for classification.

5.2 Related Works

Sparse representation represents each video by a set of independent features. Each one of these features usually captures the characteristics of a small neighborhood in the spatiotemporal space of the video. A vector of the same size often represents these features, but the number of these features might be different for each video. Hence, in sparse representation, each video will be represented by a set of independent features. To be used by any existing learning method, these features should be mapped into a vector of fixed-size. Initially, the bag of words (BOW) have been used for this purpose, but recently, the Fisher Vector (FV) encoding is used as it outperforms the BOW by a good margin [66]. Finally, an SVM [65] or a similar learning method can be used for classification.

Traditionally, sparse feature points are extracted in two steps. The first step, called feature extraction, is used to find the locations (spatial and temporal) in the video that are interesting. Having these interest points, the second step, called feature descriptor extraction, aims at encoding the information in the video around these feature points. These descriptors should be as discriminant as possible.

5.2.1 Feature Extraction Methods

In 2D still images, feature points are defined as points that carry more texture information. These are points with severe changes in the intensity. From the visual point of view, this could be a boundary line. Corners have even more information as they have intensive changes in both directions. This simple idea leads to several corner detectors algorithms to find interest points (e.g. Harris interest point detector). In the video paradigm, the same idea inspired the invention of many algorithms.

Laptev et al. [17, 41] used the idea behind standard Harris interest point detector and extended it to the 3D spatiotemporal space. Laptev et al. looked for significant changes in both spatial and temporal domain. In other words, they looked for spatial corners with meaningful motion (non-constant motion) [41].

The spatiotemporal corners are very robust feature points, but they are rare in videos. Furthermore, these interest points do not occur in all kinds of human activity videos. Dollar [19] mentioned these limitations and introduced another local feature point extractor named cuboid. The cuboid feature points are extracted by using a response function. The authors mentioned that cuboid locations and types should be sufficient for many recognition tasks.

Another approach that interestingly produced a comparable result is the random selection of interest points. Instead of searching videos for interest points, Scovanner et al. proposed to use random points as interest points [23].

Trajectory features are slightly different from traditional appearance-based feature points, as the only critical component for them is the frame and the spatial location in which they are starting. Only a few works have been done on trajectory descriptors. One of the significant works on trajectories was proposed by Wang et al. [22].

In particular, they have shown that dense sampling could outperform other methods. The dense sampling is straightforward to implement and is fast to calculate. These methods usually put a grid on top of each frame; then the grid corners are examined to find out how much visual information exists in each area. If these grid points happen to be on a smooth area of an image (no texture), they are not suitable for tracking, and they are removed. This method usually produces lots of feature points, which increases the computation time of next steps.

Some methods might provide more interest points compared to others. As a rule of thumb, more interest points means more accurate results, but also more CPU time for further processing. OpenCV implementation of FAST [61, 67] corner detector is used for finding the initial locations of trajectories. FAST corner detector is a simple corner detector, which is based on a machine learning method and selects points that are brighter or darker from the majority of their neighborhoods.

5.2.2 Feature Description Methods

Different sparse feature descriptor methods have been proposed in the literature. This section only covers the most important ones. Namely, HOG and HOF, proposed by Laptev [17, 18], Motion Boundary Histograms, proposed by Dalal et. al. [68] and Trajectory shape descriptors, proposed by Wang [21].

Histogram of Oriented Gradient (HOG) and Histogram of Optical Flow (HOF), are two popular methods of sparse feature extraction [18], where HOG captures the visual appearance information and HOF captures the dynamics of an action in consecutive frames. To calculate HOG, a dense grid is put around each interest point first. Then, a local histogram of orientation for each cell is calculated separately. By combining these histograms, the ultimate HOG feature descriptor is obtained. To make a more robust descriptor, a contrast normalization is usually done on overlapping blocks (spatial regions larger than individual cells). HOF is very similar to HOG but, instead of appearance information, optical flow information is used to build the histogram.

Dalal [68] proposed to calculate the optical flow derivatives for horizontal and vertical directions separately. This leads to MBHx and MBHy components, which will be combined later. Wang et al. [21] argued that the motion captured by HOF may come from different sources (object motion v.s. camera motion), so the camera movement can degrade the accuracy of HOF. They proposed and showed that since MBH uses derivatives, the constant camera motion will be removed.

Trajectory shape descriptors proposed by Wang were compared to HOG, HOF, and MBH. Also in their tests, dense sampling outperformed other sampling methods. They showed that the trajectory aligned HOF, HOG, and MBH could beat other descriptors, including the trajectory shape descriptor.

5.3 Proposed Method

The shape descriptor proposed by Wang is a simple but effective one. To better formulate the trajectory shape, let us define the trajectory.

Trajectories are trails of 2D spatial feature points in time (Figure 5.3.1). Any feature detector algorithm could be used to extract these 2D points. Formally, a trajectory T_k is an ordered list of spatial locations, in $l + 1$ consecutive frames, where l is called the length of the trajectory.

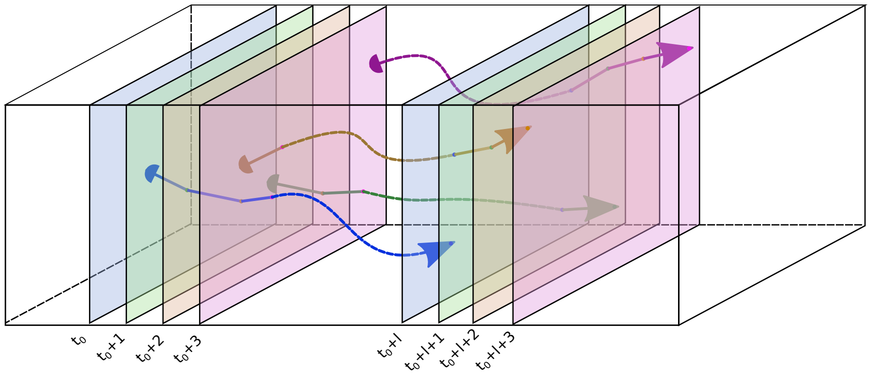

Figure 5.3.1: Tracking of feature points for l consecutive frames in spatiotemporal space

$$T_k = (p_0, p_1, p_2, ..., p_l), p_i \in \mathbb{R}^2, i = 0...l \tag{5.3.1}$$

In other words, each trajectory is simply the trails of a spatial point (feature point) in time. Each trajectory can be denoted by a function in a small domain. Let us define $T_k : time \to \mathbb{R}^2$ where $T_k(i) = p_i$. In fact, this function pinpoints the feature point location in relative time, so it is a 2D position. If we differentiate it with respect to time, we get the instance velocity at that particular moment of that particular trajectory. Since the measurements have been done in discrete times (frames), the differentiation should be defined in discrete space. We have used the forward difference for this matter.

$$V_k(t) = \frac{dT_k}{dt} = \lim_{t \to 0} \frac{T_k(t + dt) - T_k(t)}{dt} \tag{5.3.2}$$

The smallest value for t is the difference between two consecutive frames (usually between 30ms and 40ms). Since the videos usually have a fixed frame rate, we can measure the time in frames, instead of seconds. This change of unit makes our calculations easier. Hence, the smallest value for t mapped to one, making $dt = 1$ and we can write:

$$V_k(t) = T_k(t + 1) - T_k(t) \tag{5.3.3}$$

Alternatively, we may write all the values of the velocity function as:

$$V_k = (v_0, v_1, ..., v_{l-1}), v_i = p_{i+1} - p_i, p_i = T_k(i), i = 0...l - 1 \tag{5.3.4}$$

Let V_k be the velocity representation of the trajectory T_k. The following equation shows the shape descriptor proposed by Wang as defined in their paper [52].

$$W_k = \frac{(w_0, w_1, ..., w_{l-1})}{\Sigma_i \|w_i\|}, w_i = p_{i+1} - p_i, p_i = T_k(i), i = 0...(l-1) \tag{5.3.5}$$

where $\|.\|$ denotes the $L^2\ norm$. As it can be seen, the w_i's are simply defined the same way as instant velocities, i.e., $v_i = w_i$. So from (5.3.4) and (5.3.5), it is easy to show that:

$$W_k = \frac{V_k}{\Sigma_i \|v_i\|} \tag{5.3.6}$$

This means that the trajectory shape descriptor proposed by Wang et al. [52] can be interpreted as the normalized version of velocity that is proposed in this chapter. From the physical point of view, Wang's descriptors still have the sense of velocity.

The velocity can be differentiated one more time to obtain the acceleration.

$$A_k = \frac{dV_k}{dt} = (a_0, a_1, ..., a_{l-1}), a_i = v_{i+1} - v_i, v_i = V_k(i), i = 0...(l-2) \tag{5.3.7}$$

We call A_k the acceleration representation of trajectory T_k. In theory, one may continue to proceed with higher order differentiations. Each level of differentiation introduces possibly new information, but it also increases the effect of noise. In our experiments, we have tested up to the seventh derivative. It is expected that derivatives, after the third order, will not add much information, as human activities do not consist of very complex motions. On the other hand, after the third derivation, the effect of noise will affect the results and reduces the accuracy of the system.

Having the derivatives, we have proposed different descriptors by concatenating them. For example, the first descriptor D_1 is simply the first derivative. The second descriptor D_2 is the concatenation of first and second descriptors together, and the third one D_3 is the concatenation of D_1, D_2 and D_3 derivatives together, and so on. We have also defined the zero order derivative to be the same as trajectory minus its starting location. The zero order descriptor D_0 only contains zero order derivatives.

5.4 Experiments

We have tested our new method against our challenging dataset. Our dataset contains 27 different activities, performed by 11 different actors and each actor repeated each activity four times. The dataset recorded with two off the shelf cameras, originally intended for stereo vision, but to run tests of this chapter, we have only used one camera output only. The cameras are fixed in the everyday office environment. The dataset is challenging since it has many activities and some activities are similar from motion point of view.

The tests ran over various trajectory lengths. Fisher Vector is used to prepare data for learning and a multi-class C-SVM used for classification. The reported results obtained by N-Fold cross-validation, where N is the number of actors. In each run, we left all instances of one actor out and trained the SVM with the other actors; then the tests have been done on the left out actor.

Table 5.4.1 summarizes the results. Each row represents one encoding method, while each column is dedicated to a particular trajectory length.

Table 5.4.1: The effect of encoding on different trajectory lengths (the reported values are accuracies)

	Trajectory Length			
Method	11	13	15	17
D_0	83.24%	81.21%	83.32%	84.08%
D_1	84.50%	85.72%	84.37%	84.54%
D_2	**85.80%**	87.46%	86.48%	86.94%
D_3	85.30%	87.95%	87.07%	**89.09%**
D_4	84.84%	**88.42%**	**88.55%**	88.42%
D_5	84.08%	85.68%	88.21%	87.11%
D_6	84.79%	86.18%	86.44%	86.20%
D_7	82.10%	83.15%	84.96%	85.43%
Wang Method [21, 52]	83.78%	82.84%	84.58%	85.89%

The baseline for comparison is the last row of the table. This row represents the results obtained by method proposed in [21, 52] trajectory encoding algorithm (represented in

Equation 5.3.5). The proposed algorithm has outperformed Wang's in almost all cases. As expected, D_0 results are below the baseline. D_1, which is the non-normalized version of Wang's descriptor, performs similar or better. This means normalization of trajectory might remove some of the discriminant information. This makes sense as the magnitude of trajectories can be discriminative for some types of activities. In all cases, D_2 outperforms D_1 and in most cases, D_3 outperforms D_2.

As it can be seen, D_3 and D_4 outperforms other trajectory descriptors. For example, D_3 produces best results for length 17, with more than 89% accuracy. It is a 3.2% improvement, compared to the baseline. For trajectory length of 13, D_4 reaches more than 88.42% accuracy, which is a 5.58% improvement over the baseline. After D_4, there is no significant improvement. This is because human activities do not have (or have a little) higher order complexities. Furthermore, differentiation increases the effect of noise which results in losing the accuracy.

Table 5.4.2: *Dense sampling* versus *sparse sampling*

	Trajectory Length			
Method / Encoding	11	13	15	17
Dense Trajectory / Wang's Encoding	89.54%	88.90%	88.74%	87.80%
Sparse Trajectory / Proposed Encoding	85.80%	88.42%	88.55%	89.09%

We also compared our method with dense sampling results as well. Dense sampling uses much more sampling points, compared to our sparse sampling technique. The best results we obtained by our method compared with dense sampling results are presented in Table 5.4.2. As can be seen, with new trajectory encoding algorithm, we can obtain competitive results with the ones using dense trajectories. Note that this is not a fair comparison as the number of dense trajectories are higher than the sparse trajectories and as a result, they need more CPU time, while sparse trajectories are lighter and faster.

5.5 Conclusion

This chapter has reviewed different encoding methods used in the literature for human activity recognition. It examined the trajectory shape encoding method more intensively and proposed a new encoding method, which is a superset of the existing trajectory shape encoding. It also proposed to define the trajectories as functions and formally defined the new descriptors. Moreover, it also provided rational reasons to justify that this new encoding method is a superset of the existing methods, and why it should provide more accurate results.

Finally, the experiments have shown that the new encoding method outperforms the existing encoding method with a good margin. It also demonstrated that the new encoding method applied to sparse sampling could achieve competitive results to dense sampling method with fewer computations.

CHAPTER 6

Disparity-Augmented Trajectories for Human Activity Recognition

6.1 Introduction

The disparities can boost the performance of trajectory-based methods. To calculate disparities, two slightly different views of the subject are required. This chapter employs the trajectory extraction algorithm proposed in Chapter 4 to extract trajectories from left and right view. Then, these trajectories are matched against each other. The matched trajectories are mapped to a rectified image plane, where the disparity between them is calculated and fused with the actual trajectory, to create *disparity-augmented trajectory* (DAT). Finally, the trajectory shape descriptors proposed in Chapter 5 is used to encode the shape of DATs. Figure 6.1.1 demonstrates this process in a block diagram.

This chapter also improved the performance of the proposed method by limiting the

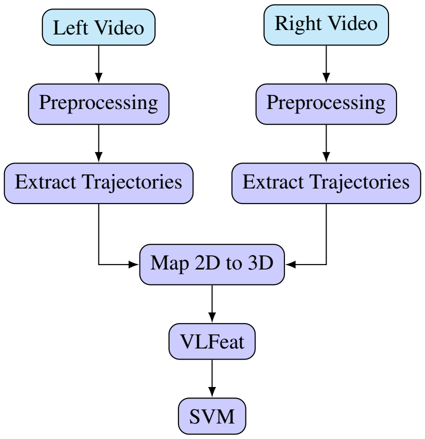

Figure 6.1.1: The block diagram of the stereo vision system

processing to the regions of interest, instead of the whole images. Our regions of interest consist of the parts in the video frames that contain movement. In particular, the graph connected component analysis algorithm can select the active areas in frames.

Note that in theory, it is possible to adequately calibrate the cameras, obtain pixel disparities, reconstruct the trajectories in the 3D space and, then perform HAR on the reconstructed trajectories. However, it is challenging in practice to keep a pair of cameras fully calibrated, because of autofocusing and other issues.

On the other hand, pixel disparities carry some 3D information and are relatively effortless to obtain. This chapter demonstrates that adding the disparity information to the 2D trajectories, can be beneficial for human activity recognition. In particular, disparity-augmented trajectories have improved classification rates by 3.11% in our tests.

This chapter is organized as follows: Section 6.2 reviews related works. Section 6.3 describes the method used for detecting human activity areas in video frames. Section 6.4 provides details for three different algorithms we have used to extract 2D trajectories and describes how the disparity information is added to the 2D trajectories. Section 6.5 describes the proposed trajectory shape encoding algorithm. Section 6.6 and Section 6.8 presents the experimental results and the conclusion, respectively.

6.2 Related works

Trajectories have proven to be useful for aligning consecutive frames before extracting low-level features [52]. Even the extraction of deep learning feature vectors benefited from trajectory alignment [55]. Trajectory shapes can also be used directly for human activity recognition.

Wang et al. [21, 22, 52] exploited trajectories in separate contributions. In their works, a grid was used to dense sample video frames. Eigenvalues of the autocorrelation matrix was utilized to filter out the samples that were not easy to track. Dense optical flow field, proposed by Farnback [69], was applied to track these sample points in time. This flow was then employed to align the interest points neighborhoods before calculating the two features, HOG and HOF. They also proposed a trajectory shape descriptor, which did not outperform the other two.

Matikainen et al. [50] used the technique of Kanade Lucas Tomasi (KLT) [70] to track a number of points and created a trajectory for each of these points. Then, they used K-Means method to cluster the obtained trajectories in different clusters (words). They have also proposed to augment these trajectories by adding some affine transformation information, which represents the motion of various parts of the body. Finally, they have used a standard bag of words (BOW) method and SVM for clustering.

In another similar work, Messing et al. [51] used KLT to track keypoints of a video and created a generative model on the velocity history of these keypoints.

Sun et al. [53] proposed to track Scale Invariant Feature Transform (SIFT) points. They have suggested using SIFT descriptors to match each keypoints across frames. They have extracted features at different levels and used multichannel nonlinear SVM for human activity recognition.

Recently, [33, 34] demonstrated that good sparse trajectories could produce competitive results to low-level features, but with less computation. Besides, trajectories are a better choice for HAR as they encode the motion of a body, while low-level features usually encode the texture or movement in small neighborhoods in spatiotemporal space. This makes low-level features more dataset dependent.

6.3 Preprocessing

We have implemented a simple, yet effective method for detecting the regions of interest (moving parts in the videos) to reduce the overall processing time. The following steps describe how we detect and remove static or stable regions from videos.

1. Estimate background with a mixture of Gaussian

2. Subtract estimated background from current frame

3. Highlight the moving parts of video by erosion and dilation operations

4. Extract the contours of motion

5. Find rectangular regions of interest as follows:

    (a) Find a bounding box for each contour

    (b) Create a graph where each node represents a bounding box and, if two boxes overlap or are close enough, have an edge between them

    (c) Use connected component labeling algorithm similar to the one in [71] to find the connected components of this graph

    (d) Combine the boxes of each connected components. Each combined box represents a separate region of interest

The above algorithm allows the extraction of all nonstatic (motion) areas.

6.4 Trajectories for Human Activity Recognition

Trajectories are defined as the trail of 2D or 3D spatial feature points in time. The disparity-augmented trajectories are similar to 3D trajectories except that they have the disparity in addition to 2D information. Formally, a trajectory is defined as an ordered list of locations, sampled over $l + 1$ time steps, where l is the length of the trajectory. In a single video, for example, the frame rate determines the distance between sampling times. So, a trajectory T in dimension n can be defined as:

$$T = (p_0, p_1, p_2, ..., p_l), p_i \in \mathbb{R}^n, i = 0...l \tag{6.4.1}$$

Note that throughout this paper we assume that $n \in \mathbb{N}$, but in practice and in our tests, $n \in \{2, 3\}$.

To create a disparity-augmented trajectory, the corresponding 2D trajectories from two views of a subject are extracted and combined. Section 6.4.1 provides more details of how these 2D trajectories are extracted. Then, Section 6.4.2 explains disparity is added to our 2D trajectories.

6.4.1 2D Trajectory Extraction

A 2D trajectory T is an ordered list of 2D spatial coordinates $p = (x, y)$ in $l+1$ consecutive frames, formally defined as:

$$T = (p_0, p_1, p_2, ..., p_l), p_i \in \mathbb{R}^2, i = 0..l \tag{6.4.2}$$

Authors in [33] compared three different trajectory extraction algorithms and showed that a combination of FAST corner detector and Farnback optical flow for trajectory extraction outperformed other trajectory extraction algorithms. More details is given below for each of the three methods.

Interest Point Tracking

This algorithm tracks feature points in time based on their appearance. We refer to this method as *“Interest Point Tracking”* (IP).

First, the interest points of the video are extracted and a local descriptor is defined for each of them. Starting from the first frame, the interest points are tracked across frames to make trajectories. When a trajectory reaches a length of $l + 1$, it is considered a complete trajectory. The full description of this process is given in Algorithm 1.

To better understand the algorithm, some definitions are given here. For each frame I_t, a set of feature points is defined as:

$$P_t = \{p | p \in \mathbb{R}^2\} \tag{6.4.3}$$

Although any image point could be considered as an interest point, we are interested in points that are easy to track across frames, for example, Harris corners.

For each member of P_t, a feature descriptor is calculated based on the appearance of the interest point neighborhood. Again, the algorithm could consider any feature descriptor.

We define a mapping $\Psi_t : P_t \to \mathbb{R}^k$, where

$$\Psi_t = \{(p, v) | p \in P_t, v \in \mathbb{R}^k\} \tag{6.4.4}$$

Algorithm 1 IP Trajectory Extraction

```
    Input the video
    Output the set of trajectories
procedure IPTrajectoryExtraction(video)
    $T \leftarrow \{\}, \tau \leftarrow \{\}$        ▷ $T$ has incomplete and $\tau$ has completed trajectories
    for $\forall I_t \in video$ do        ▷ For all frames of video
        $P_t \leftarrow ExtracFeaturesPoints(I_t)$        ▷ Extract IP's of current frame
        for $\forall tr \in T$ do        ▷ For all incomplete trajectories
            $lp \leftarrow tr\ last\ point$        ▷ Note that: $lp \in P_{t-1}$
            $N_t(lp) \leftarrow FindNeighborsInCurretnFrame(P_t, lp)$
            $bm = FindBestMatch(lp, N_t(lp))$
            if $Could\ not\ find\ the\ best\ match$ then
                $T \leftarrow T - \{tr\}$        ▷ removes $tr$ from $T$
            else
                $Add\ bm\ to\ the\ end\ of\ tr\ trajectory$
                $P_t \leftarrow P_t - \{bm\}$        ▷ removes $bm$ from $P_t$
                if len(tr) = l then        ▷ If trajectory is completed
                    $T \leftarrow T - \{tr\}, \tau \leftarrow \tau \cup \{tr\}$        ▷ move $tr$ from $T$ to $\tau$
                end if
            end if
        end for
        $\forall p_i \in P_t\ tr = CreateNewTrajectoyFrom(p_i), T = T \cup \{tr\}$
                                        ▷ Create new trajectories starting at $p_i$
    end for
    return $\tau$        ▷ Return all completed Trajectories
end procedure
```

t is the frame number, v is the feature descriptor vector for point p and k is the dimension of the descriptor (e.g., for standard SIFT descriptor $k = 128$).

The neighborhood of a point p_i is given by

$$N_t(p_0) = \{p \mid p \in P_t,\ \Delta_1(p, p_0) \leq \lambda_1\} \tag{6.4.5}$$

where $\Delta_1(.)$ is a distance measure and λ_1 determines the radius of neighborhood. In our implementation, we have used the Manhattan distance.

The best match for p_0 is found within the neighborhood $N_t(p_0)$, based on the appearance of descriptors. To do so, for each $p_i \in P_{t-1}$, a mapping $M_t(p_0) : N_t(p_0) \rightarrow \mathbb{R}$ is defined as follow:

$$M_t(p_0) = \{(p, d) | p \in N_t(p_0), d = \Delta_2(\Psi_{t-1}(p), \Psi_t(p_0))\} \tag{6.4.6}$$

where $\Delta_2(.)$ is a distance measure in descriptors space.

In our case, we used the Euclidean distance, where the closest match is considered the best match.

$$BestMatch = \arg\min\{M_{t+1}(p_i)\} \tag{6.4.7}$$

It is possible that more than one point from previous frame are a match for a single point in the current frame. In order to keep the reliable matches, we removed such points. In addition, if the similarity difference between the first and second best matches is small, the match is not robust and is thus removed. An example of the resulting trajectories is displayed on Figure 6.4.1 left (best seen in color).

Lucas-Kanade Feature Point Tracking

We refer to this method as *"Lucas-Kanade Trajectory"* (LK), as it is based on Lucas-Kanade optical flow algorithm [64]. First, the feature points of each frame of the video are extracted. Then, Lucas-Kanade optical flow algorithm is used to find the location of each of these feature points in the next frame. Based on tracking information, a trajectory will

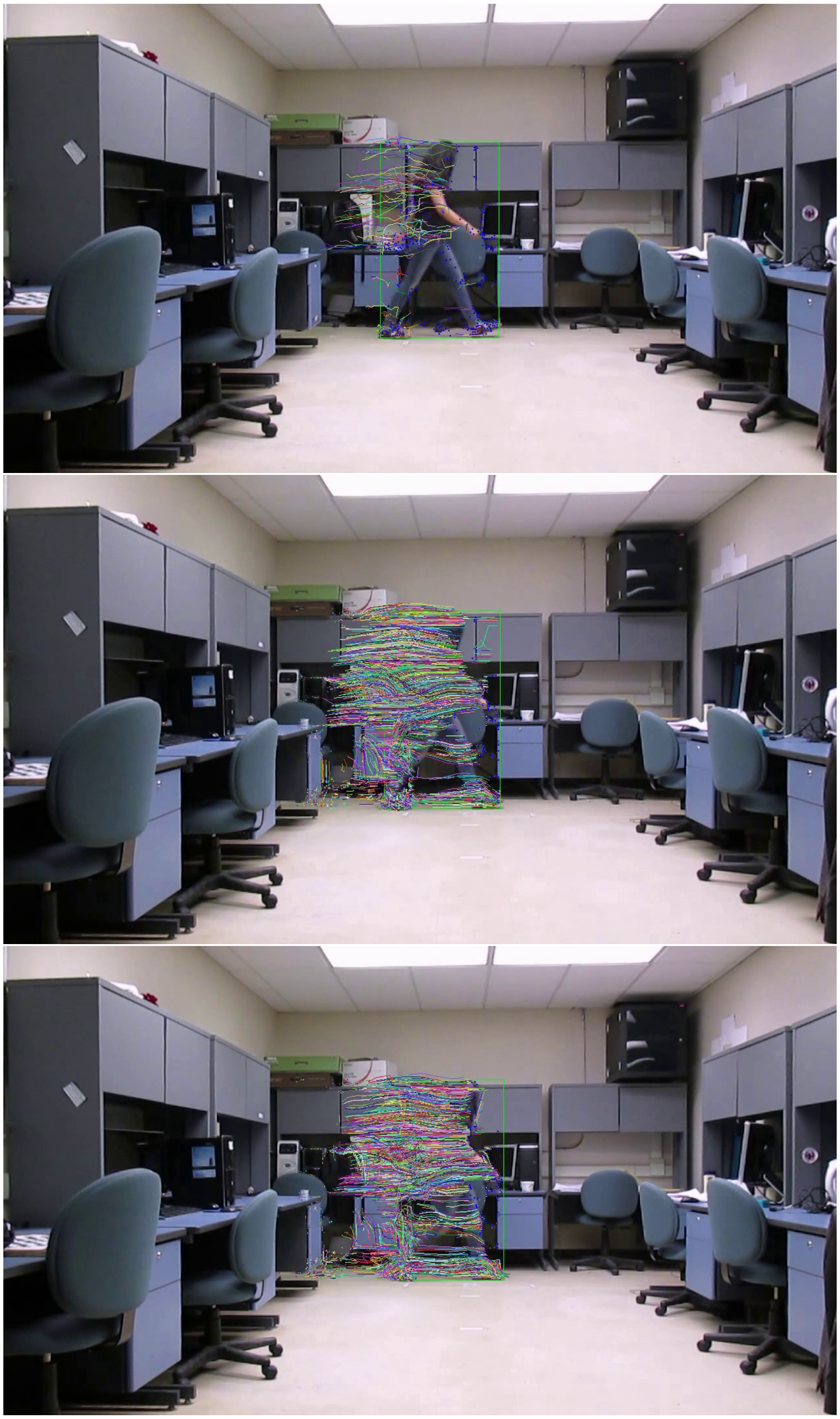

Figure 6.4.1: Three different 2D trajectory extraction algorithms (best seen in color) : IP (left), LK (middle), FB (right). The green box is the active detected area by preprocessing algorithm.

be created. The full details are given in Algorithm 2.

Algorithm 2 LK Trajectory Extraction

```
    Input the video
    Output the set of trajectories
procedure LKTrajectoryExtraction(video)
    $T \leftarrow \{\}, \tau \leftarrow \{\}$    ▷ $T$ has incomplete and $\tau$ has completed trajectories
    for $\forall \tau_t \in video$ do    ▷ For all frames of video
        $P_t \leftarrow ExtracFeaturesPoints(I_t)$    ▷ Extract Feature points of current frame
        $CP = \{\}$
        for $\forall tr \in T$ do
            $CP = CP \cup lastPoint(tr)$
        end for
        $CP = CP \cup P_t$
        $NP = FindNextPointsWithLucasKanade(CP, I_t, I_{t-1})$
        for $\forall p \in NP$ do
            if $previous\ point\ of\ p\ is\ part\ of\ a\ trajectory$ then
                $Add\ p\ to\ that\ trajectory$
            else
                $Create\ New\ Trajectory\ Starting\ at\ p$
            end if
        end for
    end for
    return $\tau$    ▷ Return all completed Trajectories
end procedure
```

Since the details of LK algorithm and Farnback feature point tracking are very similar, we explain only the latter in the next section. An example of the resulting LK trajectories are displayed on Figure 6.4.1 (middle).

Farnback Feature Point Tracking

This algorithm is based on the Farnback optical flow algorithm [69], we call it *"Farnback Trajectory"* (FB). Farnback optical flow algorithm is newer than Lucas-Kanade algorithm, and it has shown better performance [52, 69]. Farnback optical flow algorithm is also able to provide the dense optical flow field, while Lucas-Kanade optical flow was designed to track sparse feature points. This gives FB an advantage, especially when the selected points are not good feature points. Our tests revealed that FB yields competetive results to LK, and both performed much better than IP.

First, the Farnback optical flow field is calculated before the feature points of each frame are extracted. Starting from the first frame, the location of each point in the next frame is predicted, thanks to the optical flow field. These points are then connected as a trajectory. When each point is tracked in $l+1$ frames, the trajectory is considered complete. Algorithm 3 provides the details of this method.

Algorithm 3 FB Trajectory Extraction

```
    Input the video
    Output the set of trajectories
procedure FBTrajectoryExtraction(video)
    $T \leftarrow \{\}, \tau \leftarrow \{\}$    ▷ $T$ has incomplete and $\tau$ has completed trajectories
    for $\forall I_t \in video$ do    ▷ For all frames of video
        $P_t \leftarrow ExtracFeaturesPoints(I_t)$    ▷ Extract feature pints of current frame
        $OF_t = Farnback(I_t, I_t + 1)$
        $TP = P_t$    ▷ $TP$ is going to keep the point to track
        for $\forall tr \in T$ do    ▷ Add the last point of all active trajectories
            $TP = TP \cup lastPoint(tr)$    ▷ to the tracking set
        end for
        $NP = FindTheCorrespondinPoints(TP, OF_t)$
        for $\forall p \in NP$ do    ▷ Combine new found points with existing trajectories
            if $previous\ point\ of\ p\ is\ part\ of\ a\ trajectory$ then
                $Add\ p\ to\ that\ trajectory$
            else
                $Create\ New\ Trajectory\ Starting\ at\ p$
            end if
        end for
    end for
    return $\tau$    ▷ Return all completed Trajectories
end procedure
```

where, OF is the optical flow, and OF_t is its corresponding frame.

The interest points of each frame are extracted using OpenCV implementation of FAST algorithm [61]. The dense optical flow field is obtained using Farnback motion estimation method [69]. The trajectories created by this algorithm are shown on Figure 6.4.1 (right). All interest points in first frame are considered active tracking points. Their locations in the next frame are estimated by the optical flow field. The interest points in the next frame, that are not already being tracked, are added to the list of active tracking points. Once each trajectory in the active tracking set reaches the length $l+1$, it is removed from the set and added to the complete trajectory set. Trajectories created this way are shown on Figure 6.4.1.

6.4.2 Disparity-Augmented Trajectory

After the extraction of the 2D trajectories from the left and right videos, matching the trajectories is achieved based on local descriptors (Section 6.4.2). The matched trajectories are then mapped to their corresponding rectified planes (Section 6.4.2). Finally, their disparity is fused with the 2D spatial information (Section 6.4.2).

Finding Matching Trajectories

Each trajectory starting point, in the left and right videos, are encoded with a SIFT descriptor and the best match of this descriptor is found by using the algorithm from [43]. Starting from the first frame of the video, for each descriptor in the left frame, its best match is found in the right frame. To make the matching robust, we repeat the process between the right to the left frames and, as we only keep the matches that works both ways.

Video Rectification

In this step, we estimate the Fundamental Matrix F, relating left and right frames, and the rectification matrixes H_l and H_r [72]. The rectification is the process of mapping an images to a plane, where the y disparity becomes zero and only the x disparity remains. If p and p' represent two matching points, between the left and right images, the fundamental matrix F is the matrix that satisfies:

$$pFp' = 0 \tag{6.4.8}$$

The eight-point algorithm is used to estimate F [73]. The FAST algorithm is used to find the feature points. The same algorithm, as explained in Section 6.4.2, is used to match the feature points between the left and the right video frames.

The calculation of F and rectification matrices highly depend on the quality of matched points. To address these issues, we propose the following technique to find the best estima-

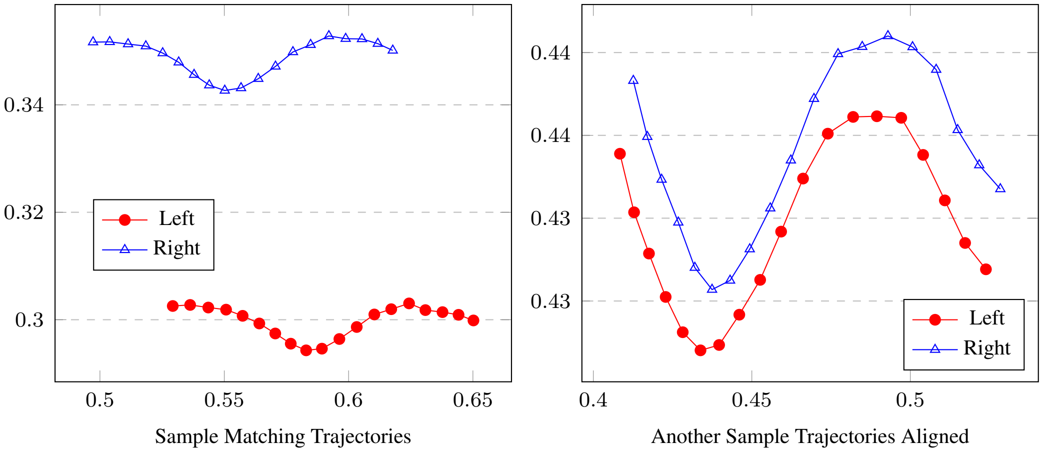

Figure 6.4.2: Sample stereo trajectories. Left image shows a sample trajectory as seen in the left and right views. Right image shows another stereo trajectory rectified by the proposed method (best seen in color).

tion of F, H_l and H_r. First, m random frames of the stereo video are selected. For each pair i of these stereo frames, F^i, H_l^i and H_r^i are calculated. If $p = (x, y, 1)^T$ and $p' = (x', y', 1)^T$ represent a matching point, then $q = H_l^i p = (u, v, 1)^T$ and $q' = H_r^i p' = (u', v', 1)^T$ represents the mapping of these corresponding points on the rectified plane, where ideally $v - v' = 0$. Considering the matched trajectories from Section 6.4.2, the best estimate of F is the one that maximizes the number of trajectories that will be rectified with the acceptable y disparity. Figure 6.4.2 shows samples of two matching trajectories, one before rectification and the other one after rectification. In addition, because the calculation of F is susceptible to outliers, we have also used the random sample consensus (RANSAC) method to make its calculation robust.

Calculating Disparity-Augmented Trajectories

Having the rectification matrices H_l and H_r, it is now easier to calculate the rectified left and right trajectories, t_l and t_r.

$$t_l = H_l T_l \tag{6.4.9}$$

$$t_r = H_r T_r \tag{6.4.10}$$

where T_l and T_r are the homogeneous representation of the left and right trajectories, respectively.

Each column of t_l or t_r represents a rectified trajectory image plane point. Consider two corresponding points $q = (u, v, 1)^T$ and $q' = (u', v', 1)^T$ on left and right trajectories, respectively. The corresponding disparity augmented point (x_m, y_m, d) will be given by:

$$x_m = u \tag{6.4.11}$$

$$y_m = v \tag{6.4.12}$$

$$d = u - u' \tag{6.4.13}$$

6.5 Trajectory Shape Descriptor

The method we used to create a descriptor for trajectories is an extension of [34]. In simple words, we record the locations of interest points in 2D or 3D spaces, and encode them into trajectories. Traditionally, a trajectory is considered as an ordered set of points. Here we treat them as discrete functions that map time values to a point coordinates. The first derivative of this function with respect to time is the velocity of the interest point. The second derivative represents the acceleration. Higher order derivatives encode higher order motion information. The final descriptor is obtained by concatenating these derivatives. In our experiments, we have used derivatives up to the 7th order, for single views and up to 5th order for multiple views.

Formally, each trajectory, defined in Equation 6.4.1, can be interpreted as a function of time that map time values to locations in $\mathbb{R}^n$ space.

$$T : time \rightarrow \mathbb{R}^n \tag{6.5.1}$$

$$T(t) = P(\frac{t - t_0}{\Delta t}) \tag{6.5.2}$$

$$P(i) = p_i \in \mathbb{R}^n \tag{6.5.3}$$

where Δt is the time between two consecutive frames and t_0 is the starting time of trajectory T_k.

The first derivative of this function is given by:

$$V(t) = \frac{dT}{dt} = \lim_{t \to 0} \frac{T(t+dt) - T(t)}{dt} \tag{6.5.4}$$

Usually, video have a fixed frame rate, which means frames of video are sampled with a fixed interval between them. Therefore, the smallest value of dt is the distance between two consecutive frames (Δt). Changing the unit of measurement from seconds to frames makes $dt = 1$. Hence, the velocity Equation 6.5.4 can be rewritten as:

$$V(t) = T(t+1) - T(t) \tag{6.5.5}$$

$$V = (v_0, v_1, ..., v_{l-1}) \tag{6.5.6}$$

$$v_i = p_{i+1} - p_i, i = 0...(l-2) \tag{6.5.7}$$

Similarly, the acceleration is given by

$$A(t) = \frac{dV(t)}{dt} = V(t+1) - V(t) \tag{6.5.8}$$

$$A = (a_0, a_1, ..., a_{l-1}), \tag{6.5.9}$$

$$a_i = v_{i+1} - v_i, i = 0...(l-2) \tag{6.5.10}$$

Higher order functions can be defined as follows:

$$\nabla^{(n)} = \frac{d^n T(t)}{dt^n} \tag{6.5.11}$$

The actual descriptor is created by the concatenation of these derivatives. For example, descriptor D_1 is given by the velocity ($\nabla^{(1)}$), descriptor D_2 is given by the concatenation of $\nabla^{(1)}$ and $\nabla^{(2)}$ and, descriptor D_3 is given by the concatenation of $\nabla^{(1)}$, $\nabla^{(2)}$ and $\nabla^{(3)}$.

ut it will converge to zero very fast (since it is simple physical movement). Another one might calculate the degree of movement: The number of times movements can be differentiated without converging to zero (my best guess is 3).

6.6 Experiments

In addition to the results, this section provides details on the dataset we have used and on our experimental setup.

6.6.1 Dataset

Since there were no suitable stereo datasets for HAR, we had to create our own to show the effectiveness of our proposed method.

Despite the amount of work that has been done on human action recognition, there is no globally accepted definition of human action. This is especially more visible in the different datasets that have been created so far. Some actions, like walking, running and jumping, are widely accepted [74, 75]. A single person usually performs these activities, with some of them containing human and objects/environment interactions, like riding a bike or shooting a basketball [76], playing cello and mopping the floor [77]. Some researchers went beyond this and created datasets for cooking different recipes [78].

To create a dataset for HAR, one need to decide which activities are suitable for the dataset. We considered actions to be sole movements of human body, regardless of the background, environment or tools they might be using. Besides, we studied activities that contain the whole human body movement as actions. We have considered 27 different actions for this dataset. The activities were selected based on the frequency of their appearance in other human datasets. Besides, we added activities that can be performed in an office setting. We did not use tools much. For example, for recording throwing, the actor does not throw anything, he/she simply acts like throwing. The only exception was for pushing/pulling of objects, where a chair has been used.

Actions were performed by eleven different volunteer actors in an everyday office setting. Some activities were performed in different scenarios. For example, walking was performed four times by each actor: walking from left to right, walking from right to left,

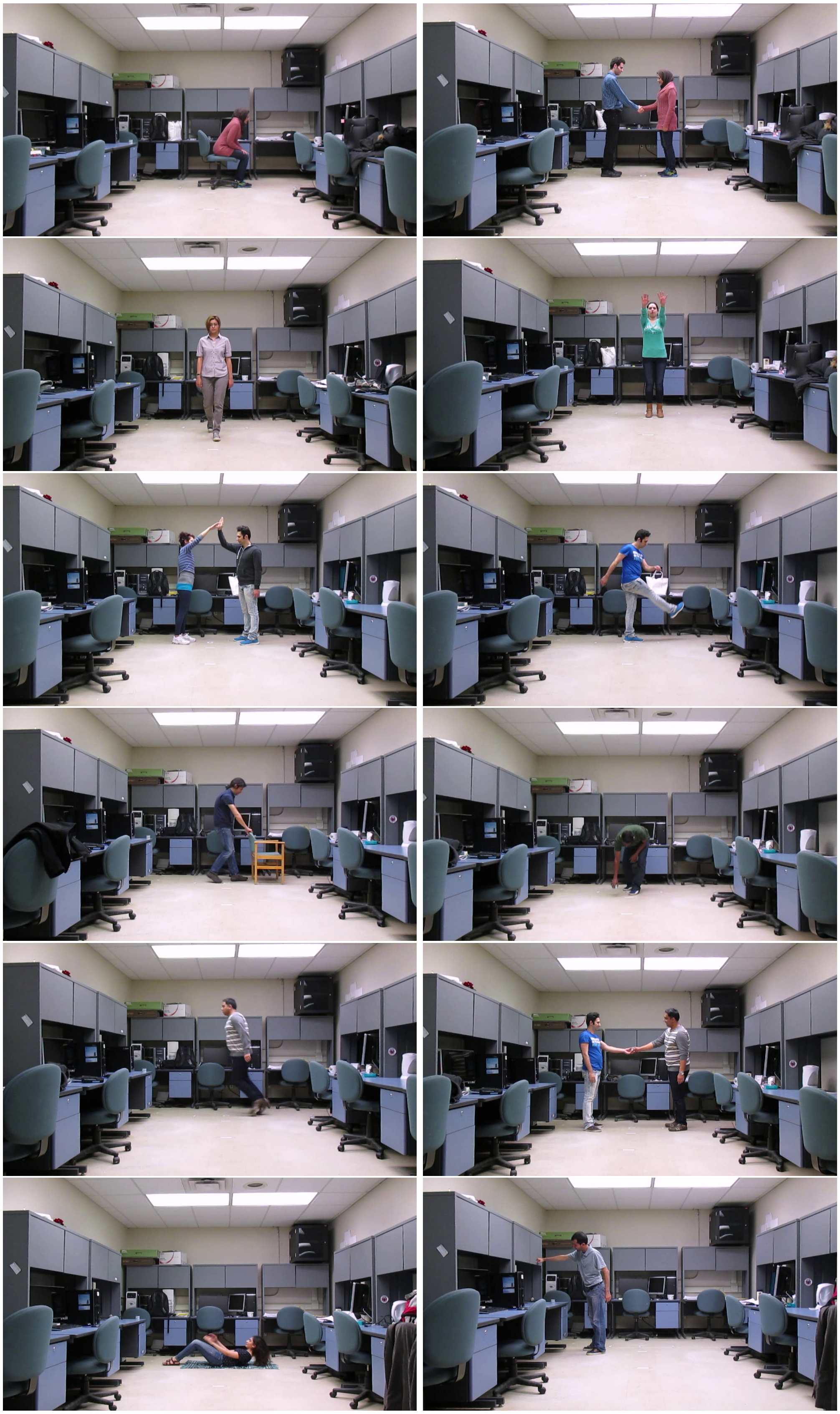

Figure 6.6.1: Sample images from dataset (best seen in color)

Table 6.6.1: The list of activity classes in the proposed dataset

Crossing Arms	Exchange Object	Hand Clapping
Hand Shaking	Hand Waving	High Five
Hitting	Jumping Over Gap	Jumping Jack
Kick the Ball	Kicking	Lay Down
Pickup(Floor)	Pull	Pointing
Pickup(Table)	Push	RaiseHand
Running	Scratch Head	Sitdown
Situp	Skipping	Standup
Throwing	Turning	Walking

walking toward the camera and walking away from the camera. We used off-the-shelf cameras to record these activities. The stereo cameras were placed roughly at 30cm from each other. The camera and background were stable during each session of recording. In total, five hours of activities were recorded by each camera. A total of 4076 stereo video clips were extracted, from which 1188 were selected to represents 27 different activities. The latter were performed four times by each of the eleven actors. Table 6.6.1 presents information about the recorded activities. Some samples of these activities are presented in Figure 6.6.1.

6.6.2 Experimental Setup

All the tests are carried out on a Ubuntu machine, with eight 3.8 GHz cores and 8G of RAM. The video processing part, including trajectory extraction, is implemented in C++, using OpenCV library. The trajectory aligning algorithm is implemented in Python.

After obtaining a set of trajectory descriptors for each video, and since [22] has shown the effectiveness of *fisher vectors* over other methods, we used *fisher vectors* to prepare data before passing it to a standard support vector machine (libSVM [65]).

The data used for training and testing was split as follows. For each action, all videos of one actor are used for testing, while the remaining videos from other actors are used for training. A confusion matrix is calculated for each action. The blending of these matrices represents the overall confusion matrix. The accuracy, reported in the tables, is the ratio of correctly classified instanced to the total number of samples, directly calculated from the

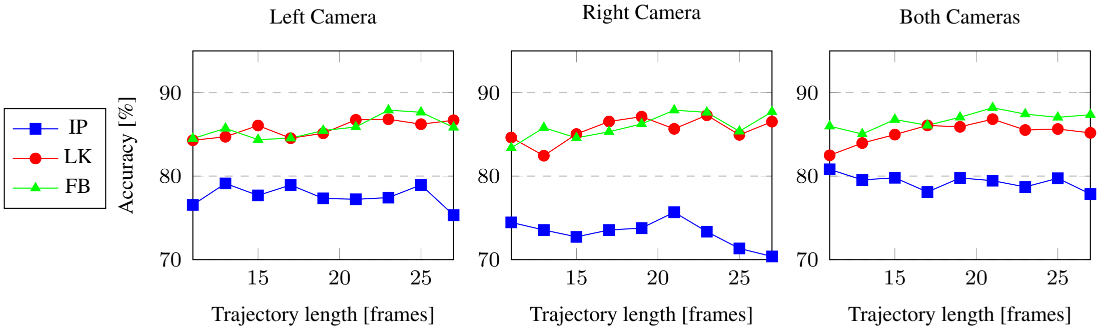

Figure 6.6.2: The perfomance of IP, LK and FB 2D trajectories on the left, right and both camera

overall confusion matrix.

6.6.3 2D Trajectories

Table 6.6.2 summarizes the obtained results of our HAR tests using 2D trajectories. Each column represents one of the algorithms proposed in this paper, where "*both camera*" column refers to the stacking of left and right descriptors without any further processing. The descriptors of the trajectories were calculated using the algorithm proposed in [21]. As it can be seen, FB outperforms the other two in most cases. LK closely follows FB and beats it in some cases. The reason why FB and LK are yielding similar results is because the selected feature points are the corners, which are easy to follow for both algorithms. Both FB and LK track pixels at subpixel accuracy, yielding smoother trajectories. On the other hand, IP algorithm uses pixel accuracy that degrades its results, as it can be seen on Figure 6.6.2.

The optimum trajectory length was 21 or 23 for LK and FB. Figure 6.6.3 compares the best results obtained from left, right and both cameras. As it can be seen, there is no significant difference between them. In other words, adding up trajectories seen by the left and right cameras did not improve the results.

Table 6.6.2: The accuracy(%) measured for left, right and both cameras. Both-camera is the combination of features from left and right frames

	Left Camera			Right Camera			Both Camera		
Length	IP	LK	FB	IP	LK	FB	IP	LK	FB
9	77.73%	83.87%	82.81%	74.75%	81.09%	82.92%	80.20%	82.89%	82.97%
11	76.55%	84.29%	84.50%	74.45%	84.62%	83.38%	**80.81**%	82.49%	85.98%
13	**79.12**%	84.71%	85.72%	73.54%	82.44%	85.80%	79.54%	83.96%	85.04%
15	77.68%	86.06%	84.37%	72.72%	85.04%	84.58%	79.80%	84.97%	86.78%
17	78.93%	84.54%	84.54%	73.54%	86.54%	85.31%	78.09%	86.07%	86.08%
19	77.33%	85.13%	85.46%	73.77%	87.12%	86.25%	79.78%	85.90%	87.05%
21	77.22%	86.74%	85.88%	**75.67**%	85.66%	**87.90**%	79.45%	**86.82**%	**88.19**%
23	77.42%	**86.81**%	**87.90**%	73.34%	**87.29**%	87.64%	78.70%	85.52%	87.43%
25	78.94%	86.22%	87.65%	71.33%	84.94%	85.37%	79.74%	85.64%	87.04%
27	75.32%	86.69%	85.84%	70.37%	86.52%	87.70%	77.85%	85.18%	87.35%

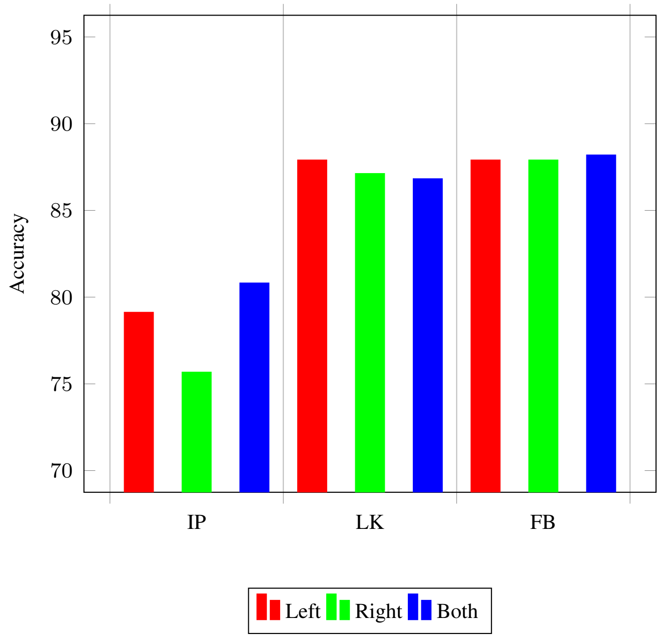

Figure 6.6.3: The accuracy of left, right and both cameras (best seen in color).

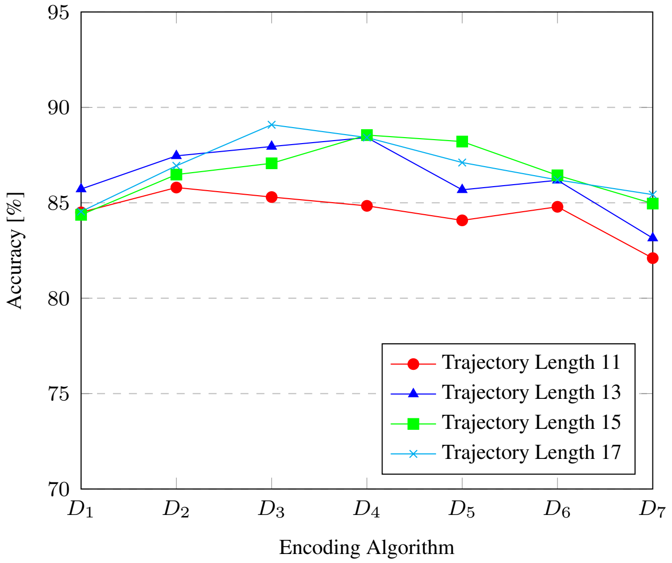

Figure 6.6.4: The effect of encoding on the accuracy of Farnback trajectories (best seen in color).

Table 6.6.3: The effect of proposed shape descriptor algorithm on the accuracy of 2D Trajectories

	Trajectory Shape Descriptor						
Length	D_1	D_2	D_3	D_4	D_5	D_6	D_7
11	84.50%	85.80%	85.30%	84.84%	84.08%	84.79	82.10
13	**85.72%**	**87.46%**	87.95%	88.42%	85.68%	86.18	83.15
15	84.37%	86.48%	87.07%	**88.55%**	**88.21%**	**86.44**	84.96
17	84.54%	86.94%	**89.09%**	88.42%	87.11%	86.20	**85.43**

6.6.4 Effect of Shape Descriptor

The proposed trajectory shape descriptor in this paper has improved the classification results. Table 6.6.3 shows the effect of the new algorithm on the accuracy. In the absence of noise, higher order derivatives might provide new information. So, higher order descriptors should produce better results in general. However, in practice the effect of noise and outliers is amplified by the derivatives. Besides, human activities do not have very complex motions. As a consequence, the accuracy has a local maximum bound. Table 6.6.3 and Figure 6.6.4 illustrate this effect. As it can be seen, D_3 and D_4 produced the best accuracy for trajectories, with length 17 and 15, respectively. As expected, higher order derivatives do not improve performance.

6.6.5 Disparity-Augmented Trajectories

Table 6.6.4 summarizes the obtained results for disparity-augmented trajectories. Each row represents a trajectory length while each column represents an encoding algorithm. We have tested trajectory lengths that range between 9 and 27, and encoding up to the fifth degree. As it can be seen, the added disparity information increased the accuracy by around 2% in all cases. The best obtained result was for trajectory length 19 and encoding degree three. The general trend is that increasing the length of trajectory increases the accuracy of the classification. This trend is more obvious on Figure 6.6.5. The best results (91.85%) were obtained with D_2 and D_3 encoding, at trajectory lengths of 21 and 19, respectively.

Figure 6.6.6 illustrates the confusion matrix of a sample test. The actual classes represented on each row and each column represent the predicted classes. The number of correct

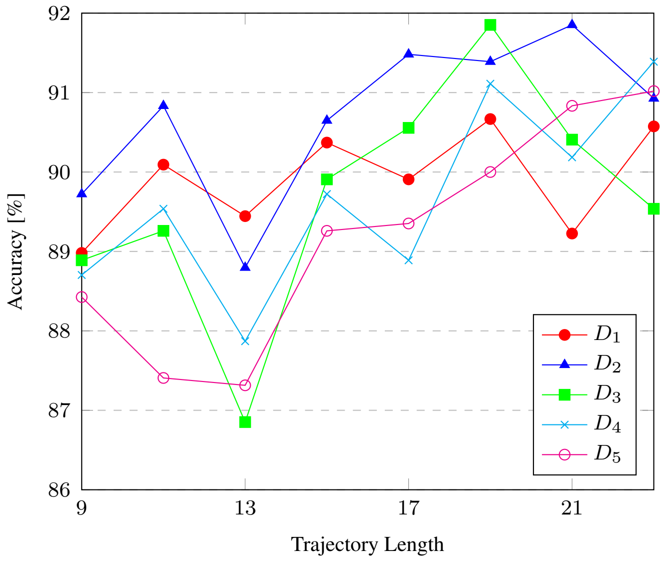

Figure 6.6.5: Disparity-augmented trajectories (DAT)

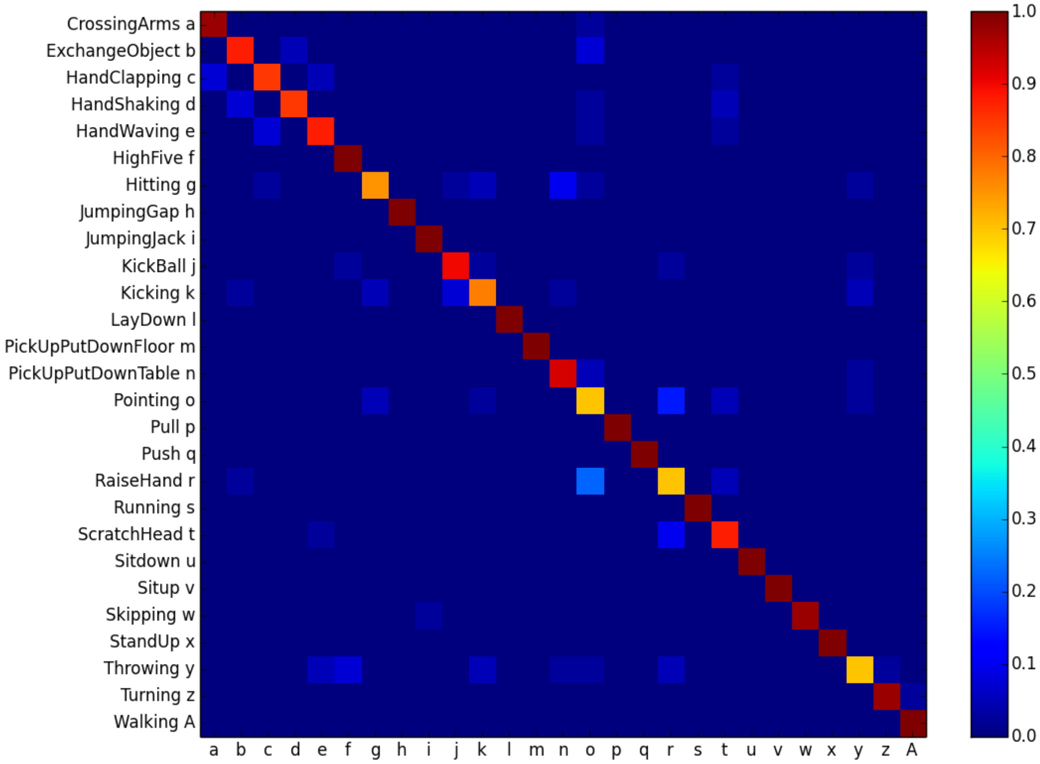

Figure 6.6.6: Sample confusion matrix for 27 classes (indexed a to z and A). Each row represents the actual class, and each column is the predicted class (best seen in color).

Table 6.6.4: The accuracy of *disparity-augmented trajectories* (DAT) on human activity recognition

	Trajectory Shape Descriptor				
Length	D_1	D_2	D_3	D_4	D_5
9	88.98%	89.72%	88.89%	88.70%	88.43%
11	90.09%	90.83%	89.26%	89.54%	87.41%
13	89.44%	88.80%	86.85%	87.87%	87.31%
15	90.37%	90.65%	89.91%	89.72%	89.26%
17	89.91%	91.48%	90.56%	88.89%	89.35%
19	**90.67**%	91.39%	**91.85**%	91.11%	90.00%
21	89.23%	**91.85**%	90.41%	90.19%	90.83%
23	90.57%	90.93%	89.54%	**91.39**%	**91.02**%

classification is normalized between zero and one. It is also worth noting that the classes in our dataset are balanced, which means the number of samples for each activity classes is the same for all classes. The misclassified instances of a matrix give interesting information about the behavior of the trajectories for HAR. For example, the most confused classes in this figure are “pointing” and “raise hand”. The fact that for pointing to something, one should raise his/her hand shows that trajectories are capable of finding this similarity, but they are unable to distinguish between them in some cases. Another example is the classes “kicking a fixed object” and “kicking the ball”, these classes have very similar motions and it is expected that these classes might be confused by any method that uses motion-based information for HAR.

From another point of view, it is also true to assume that human activities have no precise definitions. In particular, many human activities do have some overlaps. For example, someone may raise the hand to point to something or to wave. So, it is evident that there is a conceptual overlap over the definition of these classes and it is not easy to separate them conceptually.

6.7 Comparison

To the be best of our knowledge, we are the first ones to use disparity for human activity recognition. As a result it is hard to compare disparity-augmented trajectories (DAT) with

Table 6.7.1: Comparison of our method against other states of the art methods

	Method	Accuracy
Trajectory based	Sparse Trajectories	87.80%
	Dense Trajectory	88.74%
	DAT (Ours)	91.85%
Trajectory aligned	HOG	89.54%
	HOF	92.72%
	MBH	92.22%

other similar works. Table 6.7.1 shows the performance of our proposed method compared to the state of the art. The closest works to DAT are 2D dense trajectories [22] and 3D trajectories [1]. The former uses dense trajectory extraction, in contrast our proposed algorithm uses sparse feature points. Sparse methods usually produce less number of feature points compared to dense methods as a result they are faster. We applied the algorithm proposed in [52] on sparse feature points and the result reported as sparse trajectories in Table 6.7.1. As it can be seen our proposed method can outperform the dense and sparse trajectory methods with a good margin. Moreover, DAT produced better result compared to HOG, and comparable results to HOF and MBH. It should be noted that HOG, HOF and MBH are aligned by using trajectories, which means they need more computation in comparison with DAT.

Koperski et al.[1] used depth information to create 3D trajectories. They ran their tests on *MSR DailyActivity 3D* dataset, which has similar setting as our dataset, but recorded with an rgb-depth camera. It is not possible to run their algorithm on our dataset, nor we can run our method on their dataset. Just as a point of comparison, we repeated their result in Table 6.7.2. As it can be seen, they could not improve the performance of 2D trajectories by using 3D data only. They improved performance by combining 2D and 3D data. This shows the effectiveness of our proposed trajectory shape descriptor.

Table 6.7.2: 3D trajectories proposed by Koperski et al. [1].

Method	Accuracy
2D TSD	78%
3D TSD	74%
2D TSD+3D TSD	85%

6.8 Conclusion

We have presented and compared three popular trajectory-based human action recognition methods. We have also enhanced the conventional trajectory encoding algorithms by considering higher order derivatives of individual trajectories. Furthermore, we have proposed a new method based on disparity-augmented trajectories for video content analysis. Because disparities carry the scene's three-dimensional clues, we anticipated an improvement in the HAR. To the best of our knowledge, we are the first to include stereo-based disparity information in a HAR method that uses trajectories. In particular, we have fused the disparity information with motion-based features. Finally, all described methods have been evaluated on a newly created dataset, that included stereo-frame videos. We have obtained improved results, when compared to traditional trajectory-based methods.

We have also demonstrated that trajectories are useful for video content analysis in general, and for human activity recognition in particular. The proposed shape encoding algorithm has improved the accuracy of activity recognition by about 1.5%. The disparity information added to trajectories has also enhanced the results by another 2.5%.

We have also discussed some limitations associated with trajectory-based activity recognition. Activities that are similar, from the movement point of view, might be confused. We believe that some actions are conceptually overlapping and are hard to be distinguished, when using the human movement information only.

CHAPTER 7

Conclusions

Trajectories have been proven in the literature to be useful for HAR by aligning the small neighborhoods before calculating the traditional low-level features, namely histogram of Gaussian (HOG), histogram of flow (HOF) and motion boundary histogram (MBH). The trajectory shape has also been used for human activity recognition in some works, but none of them could perform as good as known descriptors, like trajectory aligned HOF.

This dissertation has investigated extensively trajectories, as mid-level features for HAR. The main focus is to improve HAR performance through a better extraction and representation of trajectories, and their augmentation with the disparity information. The latter contains the scene's 3D structure clues without requiring a full 3D reconstruction. We avoided the use of RGB-depth sensors because of their limitations and associated costs.

7.1 Contributions

The following contributions have been made by this dissertation.

1. We extensively investigated the potentials of trajectories for human activity recognition. We showed that trajectories have useful information for HAR, and we also provided the limitation of such representation.

2. We proposed three modified versions of trajectory-based methods for human activity recognition, namely interest point trajectories (IP), Lucas-Kanade based trajectories (LK), and Farnback optical flow based trajectories (FB). We evaluated the discriminant power of these algorithms, and demonstrated that the LK and FB can produce similar accuracy for HAR, if the selected tracked points represent good features.

3. We proposed a better trajectory shape descriptor extraction algorithm, and proved that it is a superset of the existing trajectory shape descriptors. We proposed to treat the trajectories as functions, this enabled us to formally define the new trajectory descriptor. We showed that the traditional trajectory shape descriptors represent the speed of trajectories, while the proposed descriptor can capture the speed, acceleration, and higher order information. The experiments demonstrated the superior performance of this method over other existing methods in the literature.

4. We proposed disparity-augmented trajectories (DAT) as a method for human activity recognition. We proved that simple disparity information can be beneficial to human activity recognition. Disparity information captures the 3D structure of the scene without the 3D Euclidean reconstruction. We fused the disparity information within the 2D trajectories and demonstrated the discriminant power of disparities for human activity recognition. To the best of our knowledge, this is the first time that disparity augmented with trajectory for HAR.

5. We created a new stereo dataset for human activity recognition, to demonstrate the effectiveness of the proposed methods. A stereo dataset of 27 different activities have been recorded and hand labelled. To the best of our knowledge, this dataset is unique in its kind.

The extensive experiments we have carried out demonstrates that our method, based on disparity-augmented trajectories (DAT), outperforms other trajectory-based methods with a good margin. We have also shown that DAT can produce results better than HOG, and competetive to HOF and MBH. Disparities can be an excellent depth clue of the scene, that are useful to HAR.

7.2 Limitations and future work

Our Extensive use of trajectories also reveals that the trajectories are not the best descriptors to differentiate between certain activities. There are human activities that have overlapping motion, which confuses any method that uses motion as the main clue. For example, scratching head and hand waving both involve raising hands. These classes are easily confused by any method that uses motion as the only clue. From another point of view, it can be argued that some activities are conceptually overlapping. For example, our classifiers, sometimes confused *pushing heavy object* with *walking*. The pushing of a heavy object, conceptually overlaps with walking, as it involves a human pushing a heavy object while walking. These examples demonstrate the limitations of motion-based activity recognition.

We can see the following potential possible improvements to our work.

- Fusing the DAT trajectories with low-level features, HOG for instance, should improve the final classification results. As motion alone is not enough to separate certain activities that are conceptually overlapping, some low-level information, especially the appearance based descriptors, can be fused with DAT trajectories to improve their classification power.

- The methods proposed in this dissertation did not cope with camera movements. If the camera is not fixed, some unwanted trajectories may appear in the scene. There are methods in the literature that can be used to model the camera motion and extract the actual human motion.

- Trajectories are relatively easy to extract and they can be used for other classification tasks as well. The methods discussed in this dissertation can be adapted and applied to any other video classification task. The only requirement is that the motion of the video is discriminant.

References

[1] Michal Koperski, Piotr Bilinski, and Francois Bremond. 3d trajectories for action recognition. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 4176–4180. IEEE, 2014.

[2] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2):201–211, 1973.

[3] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[4] Lee W Campbell and Aaron F Bobick. Recognition of human body motion using phase space constraints. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 624–630. IEEE, 1995.

[5] Md Zia Uddin, Nguyen Duc Thang, Jeong Tai Kim, and Tae-Seong Kim. Human activity recognition using body joint-angle features and hidden markov model. *Etri Journal*, 33(4):569–579, 2011.

[6] Lu Xia, Chia-Chih Chen, and JK Aggarwal. View invariant human action recognition using histograms of 3d joints. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 20–27. IEEE, 2012.

[7] Yaser Sheikh, Mumtaz Sheikh, and Mubarak Shah. Exploring the space of a human

action. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 144–149. IEEE, 2005.

[8] Cen Rao and Mubarak Shah. View-invariance in action recognition. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–316. IEEE, 2001.

[9] Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE, 1992.

[10] Hsuan-Sheng Chen, Hua-Tsung Chen, Yi-Wen Chen, and Suh-Yin Lee. Human action recognition using star skeleton. In *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 171–178. ACM, 2006.

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

[12] Abdunnaser Abdulhamid Diaf. *Eigenvector-based Dimensionality Reduction for Human Activity Recognition and Data Classification*. PhD thesis, University of Windsor, 2013.

[13] Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 984–989. IEEE, 2005.

[14] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1395–1402. IEEE, 2005.

[15] Yan Ke, Rahul Sukthankar, and Martial Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[16] Olivier Chomat and James L Crowley. Probabilistic recognition of activity using local appearance. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[17] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[18] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[19] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 65–72. IEEE, 2005.

[20] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Computer Vision–ECCV 2008*, pages 650–663. Springer, 2008.

[21] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, 2013.

[22] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3551–3558. IEEE, 2013.

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

[24] Alexander Klaser and Marcin Marszalek. A spatio-temporal descriptor based on 3d-gradients. 2008.

[25] Matteo Bregonzio, Shaogang Gong, and Tao Xiang. Recognising action as clouds of space-time interest points. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1948–1955. IEEE, 2009.

[26] Angela Yao, Juergen Gall, Gabriele Fanelli, and Luc J Van Gool. Does human action recognition benefit from pose estimation?. In *BMVC*, volume 3, page 6, 2011.

[27] Amit Kale, Aravind Sundaresan, AN Rajagopalan, Naresh P Cuntoor, Amit K Roy-Chowdhury, Volker Kruger, and Rama Chellappa. Identification of humans using gait. *IEEE Transactions on image processing*, 13(9):1163–1173, 2004.

[28] James F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[29] James F Allen and George Ferguson. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579, 1994.

[30] Eli Shechtman and Michal Irani. Space-time behavior based correlation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 405–412. IEEE, 2005.

[31] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *Communications Surveys & Tutorials, IEEE*, 15(3):1192–1209, 2013.

[32] JK Aggarwal and Michael S Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43(3):16, 2011.

[33] Boubakeur Boufama, Pejman Habashi, and Imran Shafiq Ahmad. Trajectory-based human activity recognition from videos. In *Advanced Technologies for Signal and Image Processing (ATSIP), 2017 3rd International Conference on*. IEEE, 2017.

[34] Pejman Habashi, Boubakeur Boufama, and Imran Shafiq Ahmad. A better trajectory shape descriptor for human activity recognition. In *Image Analysis and Recognition:*

14th International Conference, ICIAR 2017, Montreal, QC, Canada, July 5–7, 2017, Proceedings, pages 330–337, Cham, 2017. Springer International Publishing.

[35] A Yilmaz and Mubarak Shah. Recognizing human actions in videos acquired by uncalibrated moving cameras. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 150–157. IEEE, 2005.

[36] Jan Cech and Radim Sara. Efficient sampling of disparity space for fast and accurate matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[37] Mathieu Barnachon, Saïda Bouakaz, Boubakeur Boufama, and Erwan Guillou. Ongoing human action recognition with motion capture. *Pattern Recognition*, 47(1):238–247, 2014.

[38] Mathieu Barnachon, Saïda Bouakaz, Boubakeur Boufama, and Erwan Guillou. Human actions recognition from streamed motion capture. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3807–3810. IEEE, 2012.

[39] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1297–1304. Ieee, 2011.

[40] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, 1962.

[41] Ivan Laptev and Tony Lindeberg. Interest point detection and scale selection in space-time. In *Scale Space Methods in Computer Vision*, pages 372–387. Springer, 2003.

[42] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[43] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[44] Li-Jia Li, Hao Su, Li Fei-Fei, and Eric P Xing. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Advances in neural information processing systems*, pages 1378–1386, 2010.

[45] Sreemanananth Sadanand and Jason J Corso. Action bank: A high-level representation of activity in video. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1234–1241. IEEE, 2012.

[46] Konstantinos G Derpanis, Mikhail Sizintsev, Kevin Cannons, and Richard P Wildes. Efficient action spotting based on a spacetime oriented structure representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1990–1997. IEEE, 2010.

[47] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3192–3199. IEEE, 2013.

[48] Leonid Pishchulin, Mykhaylo Andriluka, and Bernt Schiele. Fine-grained activity recognition with holistic and pose based features. *arXiv preprint arXiv:1406.1881*, 2014.

[49] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(12):2878–2890, 2013.

[50] Pyry Matikainen, Martial Hebert, and Rahul Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 514–521. IEEE, 2009.

[51] Ross Messing, Chris Pal, and Henry Kautz. Activity recognition using the velocity histories of tracked keypoints. In *2009 IEEE 12th international conference on computer vision*, pages 104–111. IEEE, 2009.

[52] Heng Wang, Alexander Klaser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[53] Ju Sun, Xiao Wu, Shuicheng Yan, Loong-Fah Cheong, Tat-Seng Chua, and Jintao Li. Hierarchical spatio-temporal context modeling for action recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2004–2011. IEEE, 2009.

[54] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[55] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4305–4314, 2015.

[56] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[57] Gül Varol and Albert Ali Salah. Efficient large-scale action recognition in videos using extreme learning machines. *Expert Systems with Applications*, 42(21):8274–8282, 2015.

[58] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International journal of computer vision*, 79(3):299–318, 2008.

[59] Pejman Habashi, Boubakeur Boufama, and Imran Shafiq Ahmad. The bag of micro-movements for human activity recognition. In *International Conference Image Analysis and Recognition*, pages 269–276. Springer, 2015.

[60] Yun-Suk Kang and Yo-Sung Ho. Efficient stereo image rectification method using horizontal baseline. In *Advances in Image and Video Technology*, pages 301–310. Springer, 2012.

[61] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[62] Mathieu Barnachon, Saïda Bouakaz, Boubakeur Boufama, and Erwan Guillou. A real-time system for motion retrieval and interpretation. *Pattern Recognition Letters*, 34(15):1789–1798, 2013.

[63] Angela Yao, Juergen Gall, and Luc Van Gool. Coupled action recognition and pose estimation from multiple views. *International journal of computer vision*, 100(1):16–37, 2012.

[64] Jean-Yves Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[65] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[66] Yang Wang, Vinh Tran, and Minh Hoai. Evolution-preserving dense trajectory descriptors. *arXiv preprint arXiv:1702.04037*, 2017.

[67] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2010.

[68] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *Computer Vision–ECCV 2006*, pages 428–441. Springer, 2006.

[69] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. *Image analysis*, pages 363–370, 2003.

[70] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

[71] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):583–598, 1991.

[72] Richard I Hartley. Theory and practice of projective rectification. *International Journal of Computer Vision*, 35(2):115–127, 1999.

[73] Richard I Hartley. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997.

[74] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[75] Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–123. IEEE, 2001.

[76] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos in the wild. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1996–2003. IEEE, 2009.

[77] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[78] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1194–1201. IEEE, 2012.

[79] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[80] Samsu Sempena, Nur Ulfa Maulidevi, and Peb Ruswono Aryan. Human action recognition using dynamic time warping. In *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, pages 1–5. IEEE, 2011.

[81] Alexei A Efros, Alexander C Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 726–733. IEEE, 2003.

[82] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[83] Steven M Seitz and Charles R Dyer. View-invariant analysis of cyclic motion. *International Journal of Computer Vision*, 25(3):231–251, 1997.

[84] Jose M Chaquet, Enrique J Carmona, and Antonio Fernández-Caballero. A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, 117(6):633–659, 2013.

[85] Kishore K Reddy and Mubarak Shah. Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, 24(5):971–981, 2013.

[86] Marcin Marszalek, Ivan Laptev, and Cordelia Schmid. Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936. IEEE, 2009.

[87] Alonso Patron-Perez, Marcin Marszalek, Andrew Zisserman, and Ian Reid. High five: Recognising human interactions in tv shows. 2010.

[88] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2556–2563. IEEE, 2011.

[89] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. June 2014.

[90] Leonid Sigal and Michael J Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. *Brown Univertsity TR*, 120, 2006.

[91] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2):249–257, 2006.

[92] Daniel Weinland, Mustafa Özuysal, and Pascal Fua. Making action recognition robust to occlusions and viewpoint changes. In *Computer Vision–ECCV 2010*, pages 635–648. Springer, 2010.

[93] Kaiqi Huang, Shiquan Wang, Tieniu Tan, and Stephen J Maybank. Human behavior analysis based on a new motion descriptor. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(12):1830–1840, 2009.

[94] Kaiqi Huang, Dacheng Tao, Yuan Yuan, Xuelong Li, and Tieniu Tan. View-independent behavior analysis. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(4):1028–1035, 2009.

[95] Shandong Wu, Omar Oreifej, and Mubarak Shah. Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1419–1426. IEEE, 2011.

[96] Nikolaos Gkalelis, Hansung Kim, Adrian Hilton, Nikos Nikolaidis, and Ioannis Pitas. The i3dpost multi-view and 3d human action/interaction database. In *Visual Media Production, 2009. CVMP'09. Conference for*, pages 159–168. IEEE, 2009.

[97] Sanchit Singh, Sergio A Velastin, and Hossein Ragheb. Muhavi: A multicamera human action video dataset for the evaluation of action recognition methods. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 48–55. IEEE, 2010.

[98] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. 2013.

[99] Cristian Sminchisescu Catalin Ionescu, Fuxin Li. Latent structured models for human pose estimation. In *International Conference on Computer Vision*, 2011.

[100] Bingbing Ni, Gang Wang, and Pierre Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. In *Consumer Depth Cameras for Computer Vision*, pages 193–208. Springer, 2013.

[101] Robert B Fisher. The pets04 surveillance ground-truth data sets. In *Proc. 6th IEEE international workshop on performance evaluation of tracking and surveillance*, pages 1–5, 2004.

[102] José Carlos Bins Filho. Context aware vision using image-based active recognition. 2004.

[103] Scott Blunsden and RB Fisher. The behave video dataset: ground truthed video for multi-person behavior classification. *Annals of the BMVA*, 2010(4):1–12, 2010.

[104] Anh T Nghiem, Francois Bremond, Monique Thonnat, and Valery Valentin. Etiseo, performance evaluation for video surveillance systems. In *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pages 476–481. IEEE, 2007.

[105] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3153–3160. IEEE, 2011.

[106] Corey McCall, Kishore K Reddy, and Mubarak Shah. Macro-class selection for hierarchical k-nn classification of inertial sensor data. In *PECCS*, pages 106–114, 2012.

[107] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *Computer Vision–ECCV 2010*, pages 392–405. Springer, 2010.

Vita Auctoris

NAME:	Pejman Habashi
DATE OF BIRTH:	1984
PLACE OF BIRTH:	Iran
EDUCATION:	Doctor of Philosophy in Computer Science, University of Windsor, Windsor, Ontario, Canada, 2018. Master of Science in Computer Engineering - Artificial Intelligence, Sharif University of Technology, Tehran, Iran, 2008. Bachelor of Science in Computer Engineering - Software, Sadjad University of Technology, Mashad, Iran, 2006.