UNIVERSITÀ DEGLI STUDI DI PALERMO

Dottorato in Ingegneria dell'Innovazione Tecnologica
Dipartimento di Ingegneria
SSD: ING-INF/05

WATCHING PEOPLE: ALGORITHMS TO STUDY HUMAN MOTION AND ACTIVITIES

IL DOTTORE
ING. VITO MONTELEONE

IL COORDINATORE
CH.MO PROF. SALVATORE GAGLIO

IL TUTOR
PROF. ROBERTO PIRRONE

CO TUTOR
PROF. MARCO LA CASCIA
PROF.SSA LILIANA LO PRESTI

CICLO XXXII
ANNO CONSEGUIMENTO TITOLO 2020

I dedicate this thesis to my family.

Abstract

Nowadays human motion analysis is one of the most active research topics in Computer Vision and it is receiving an increasing attention from both the industrial and scientific communities. The growing interest in human motion analysis is motivated by the increasing number of promising applications, ranging from surveillance, human–computer interaction, virtual reality to healthcare, sports, computer games and video conferencing, just to name a few. The aim of this thesis is to give an overview of the various tasks involved in visual motion analysis of the human body and to present the issues and possible solutions related to it.

In this thesis, visual motion analysis is categorized into three major areas related to the interpretation of human motion: tracking of human motion using virtual pan-tilt-zoom (vPTZ) camera, recognition of human motions and human behaviors segmentation.

In the field of human motion tracking, a virtual environment for PTZ cameras (vPTZ) is presented to overcame the mechanical limitations of PTZ cameras. The vPTZ is built on equirectangular images acquired by 360° cameras and it allows not only the development of pedestrian tracking algorithms but also the comparison of their performances. On the basis of this virtual environment, three novel pedestrian tracking algorithms for 360° cameras were developed, two of which adopt a tracking-by-detection approach while the last adopts a Bayesian approach.

The action recognition problem is addressed by an algorithm that represents actions in terms of multinomial distributions of frequent sequential patterns of different length. Frequent sequential patterns are series of data descriptors that occur many times in the data. The proposed method learns a codebook of frequent sequential patterns by means of an apriori-like algorithm. An action is then represented with a *Bag-of-Frequent-Sequential-Patterns* approach. In the last part of this thesis a methodology to semi-automatically annotate behavioral data given a small set of manually annotated data is presented. The resulting methodology is not only effective in the semi-automated annotation task but can also be used in presence of abnormal behaviors, as demonstrated empirically by testing the system on data collected from children affected by neuro-developmental disorders.

Acknowledgments

I wish to thank Prof. Roberto Pirrone for being my supervisor during my PhD studies. I would like to express my deeply-felt gratitude to Prof. Marco La Cascia and Prof. Liliana Lo Presti, who thanks to their patient guidance, valuable suggestions and constant encouragement make me finishing my PhD thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When we look at a person our visual system is able to recognize which kind of posture he is performing, which kind of action he is assuming and even interpreting his behavior giving abstract meanings to body movements. These types of analysis are simple and immediate for the human visual system but the development of Computer Vision system capable of doing it is still a challenging task. The interpretation of visual data collected by recording people while they are performing different activities or interactions can involve multiple types of elaborations. Visual analysis of human motion deals with the study of body movements from image sequences and is currently one of the most active research topics in Computer Vision due to the wide spectrum of promising applications connected with it, such as surveillance, human–computer interaction, home automation, healthcare, sports, computer games, animation, video conferencing, virtual reality and automatic annotation. Visual analysis of human motion usually attempts to detect, track and identify people, and more generally, interpret human behaviors from image sequences. Thus, human motion analysis is usually categorized into three major areas: human tracking [5, 6, 7, 8, 9], activities recognition [10, 11, 12] and motion analysis of human body parts [13, 14].

Although rigorous studies have been already done in these different areas, there is still a plethora of difficulties to address in the human motion analysis problem. Abrupt motion changes, complex and cluttered environments with illumination changes, noisy data, full or partial occlusions, are just some of the problems that are needed to be addressed in human motion analysis. Moreover, a typical issue to address is the modeling of human body as a non-rigid, deformable and articulated object capable to acquire a variety of postures over time.

This thesis investigates some of such issues by detailing solutions proposed and developed during my PhD studies.

Most of the contents of my thesis was presented at international conferences and published or submitted. In particular, the algorithms presented in 3.2.1, 3.3.1 are published in [15, 16]. The algorithm presented in 3.2.2 is submitted on an international conference. The algorithm presented in 4.1 is published in [17] and the one presented in 4.2 is submitted on IEEE Transactions on Cognitive and Developmental Systems.

All the algorithms discussed and tested in this thesis have been implemented in C++ and/or Python with the use of the OpenCV library.

1.1 Contributions

The contribution of this thesis is the analysis and interpretation of human motion, with particular attention to detection, tracking, people identification and interpretation of human behaviors from videos. Methods for pedestrian tracking, action recognition and for the analysis of human behavior by means of trajectories of human body parts are presented and detailed in the thesis.

As shown in Fig.1.1 the methods presented in the thesis are not part of a single pipeline. They analyze each of the research areas mentioned before to show their limits and to suggest possible solutions.

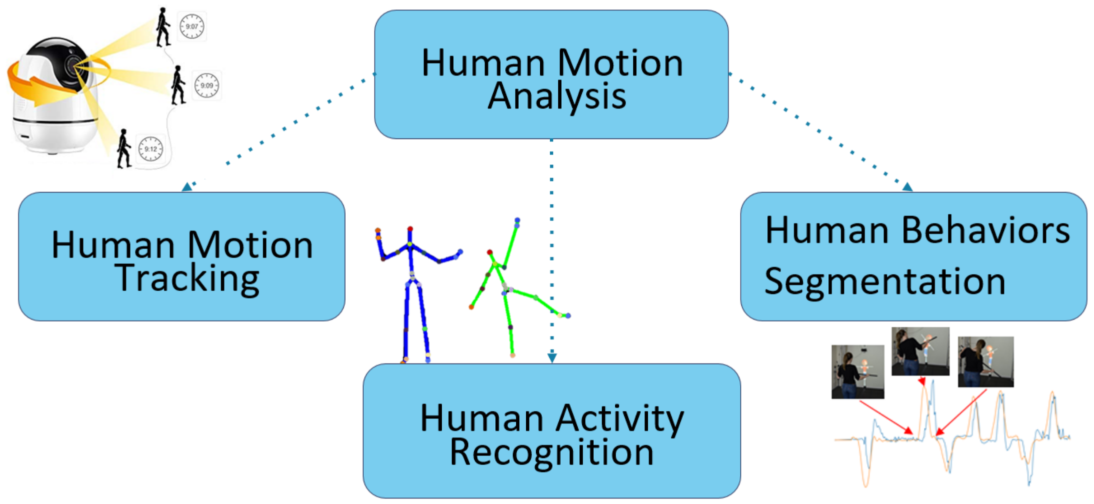


Figure 1.1: Human motion analysis is usually categorized into three major areas: human motion tracking, human activity recognition and human behaviors segmentation.

1.1.1 Pedestrian Tracking

Pedestrian tracking is the process of locating one or more persons over time by using a camera. It is commonly associated with the field of video surveillance even if it has a wide variety of uses and application fields.

Pedestrian tracking is a well-established research field when the environment and the camera are static but when the camera is active, as in the case of pan, tilt, zoom (PTZ) cameras, there are many open issues to be solved. Some of these issues concern: illumination changes, occlusion, pedestrian/object movements, people re-identification, servomotor control and cluttered background. PTZ tracking results are difficult to reproduce since the tracking is performed online by modifying the camera parameters. The difficulty to reproduce the PTZ tracking results pose considerable limitations to the development of PTZ algorithms. Thanks to the recent introduction of 360° cameras, we took advantage of the equirectangular images acquired by this type of camera to simulate a mechanical PTZ camera, called virtual PTZ camera (vPTZ). In this way our idea of developing virtual PTZ trough 360° cameras permitted not only the comparison of PTZ tracking algorithms under the same environmental conditions, but also the definition of PTZ tracking evaluation metrics. A virtual environment allows to simulate the same environmental conditions of a scenario that, otherwise, cannot be obtained. In this way, PTZ algorithms can be executed under the same environmental conditions as many times as desired, allowing tracking comparison. A vPTZ camera provides the same functional properties as a mechanical pan, tilt, zoom setup with the exception that it does not suffer from the physical limitations of the servo motors and of the mechanical delay (even if a virtual delay could be also simulated).

Our studies have mainly focused on obtaining a correct prediction of the vPTZ camera parameters, pan, tilt and zoom, to keep the tracked target at the center of its field-of-view (fov). When a target moves fast, a camera should respond quickly and accurately, and when a target moves slowly a camera should track the target smoothly with a lower speed. To solve the visual tracking problem, many considerations need to be done: dynamic motion and appearance models are just two of the many factors to consider during the development of an efficient tracking system, as well as dealing with targets that move arbitrarily in a scene, exhibiting non-linear dynamics and non-Gaussian random effects.

Experimental results of our tracking algorithms have shown that the idea of using vPTZ cameras to perform tracking in 360° video is viable. Our proposed tracking algorithms allowed to track a target in a 360° video by controlling a virtual PTZ camera. We presented different strategies to address the problem of tracking in 360° video.

In a work we modeled the target's appearance by a fixed-length dynamic memory that stores a predefined number of target color histograms. The use of the dynamic memory of the past target detections allowed to exclude false positive detections and to account for the varying appearance of the target. To decide how and when updating the dynamic memory, a threshold on the Bhattacharyya distance was also learned online. Afterwards, in another work we adopted the pre-trained deep convolutional neural network, Mask R-CNN, to strengthen the appearance model of a target by means of the binary segmentation mask of the objects detected by the network.

Furthermore, we developed a framework to track pedestrian in a 360° video by means of particle filter. In this framework the vPTZ camera was modeled as a stochastic variable. The particle filter algorithm was adopted to model the posterior distribution of the underlying stochastic process by means of a set of particles each representing a vPTZ camera view. The posterior distribution of the vPTZ cameras is approximated by a discrete set of particles. Each particle represents a vPTZ camera with specific values of pan, tilt and zoom. Such cameras are weighted and used, frame-by-frame, to estimate the vPTZ camera to track the target. To the best of our knowledge, there is a lack of public datasets for pedestrian tracking in 360° video with the exception of the dataset presented in [3]. Deep Learning-based approaches are not applicable due to this lack. Our studies on pedestrian tracking in 360° video allowed us to outperform the state-of-the-art on the 360° video dataset mentioned above, showing that our strategies are viable to obtain better results.

1.1.2 Human Behavior Characterization

The second part of the thesis presents methods to analyze humans' movements, focusing in particular on human activity recognition and behaviors segmentation, modeling them in terms of multi-dimensional time-series. These methods usually take advantage of measurements of multi-modal characteristics [18] such as gait, body poses, body movements, eye gaze and facial expressions. All these measurements can be derived from signals acquired by a wide variety of sensors (microphones, cameras, wearable sensors such as accelerometers or gyroscopes) and led to represent human behavioral responses in terms of one or more synchronized, multi-modal signals [19]. In this thesis we focused ourself in the analysis of the data acquired by visual sensors, such as rgb, depth and 360° cameras.

One of the classical approaches adopted to represent human actions is the *Bag Of Visual Words (BoVW)* [20, 21, 22, 23, 24]. In the BoVW approach, an action is represented as a distribution of image/video patches (visual words). The codebook of visual words is generally computed by clustering algorithms, i.e. k-means ([25, 26, 27, 28]). One of the main limitations of the BoVW approach concerns the lack of spatial and temporal structure information of the visual words. We addressed the problem of human action recognition by developing a descriptor, called Histogram of Patterns for Human Action Representation, in which a modified apriori algorithm was used to learn a codebook of frequent sequential patterns to represent an action in terms of histogram of frequent patterns (HoP). In contrast to the classical BoVW approach we describe an action by means of frequent sequences of visual descriptors, thus focusing more on the body motion dynamics rather than the actual body poses. Our method shown to have benefits from ignoring infrequent patterns both in terms of recall and computational complexity, since a more compact sequence description can be obtained with a smaller codebook. However, considering only the most frequent patterns may result in a lost of details of the action representation and, hence, might have a negative impact on the performance of the method.

Studies on the identification of transitions from one behavior to another in multi-dimensional time-series have also been carried out. The characterization of human behaviors in cognitive sciences provides clues to understand and describe people personal and interpersonal functioning. By assuming that humans' behaviors are represented through multi-dimensional time-series, the analysis of the observed measurements can highlight breakpoints and transitions from one behavior to another. Such change-points are a powerful tool to segment the observations and reveal events, correlations, causalities and synchronous phenomena, but also to discover abnormal behaviors or delays in the expected behavioral changes.

We considered the problem of jointly recognizing among N different classes of change-points and detecting the time when such change-points arise. We aimed at solving this problem without any a priori knowledge of the involved behavior classes. This problem is difficult because: behavior duration and type may largely change, there may be intra- and inter-subjects variations in behaviors and, in general, the transitions from one behavior to another are not abrupt. The behavioral signal was analyzed in terms of temporal windows. For each temporal window, a description of the statistical moments of the signal within the window were considered. Hence, the descriptor was fed in input to a classifier that returned the predicted change-point class. Since the annotation of humans' behaviors can be extremely expensive in terms of temporal and human resources demanded to expert annotators, our methodology is capable to semi-automatically annotate behavioral data given a small training set of manually annotated data. This method has been tested on a human behavior dataset focused on children movements. This system had not the ambition of operating as a fully automated change-point annotation system, but as a convenient, semi-automated tool for psychologists, psychiatrists, computer scientists, cognitive scientists and other practitioners working on behavior understanding, affective computing, social robotics or, more in general, human-machine interaction, that need an efficient tool for annotating behavioral observations fast.

1.2 Thesis Outline

The outline of the thesis is as follows. Chapter 2 presents an overview of the state-of-the-art in human motion analysis. Human motion analysis is presented as the macro area to which human motion tracking, human activity recognition and motion analysis of human body parts belong to.

In Chapter 3, pedestrian tracking for PTZ cameras are discussed. A virtual environment for PTZ cameras is presented together with three pedestrian tracking algorithms for 360° cameras.

In Chapter 4, methods to analyze human movements for human activity recognition (HAR) and for human behavior understanding, are discussed. In Chapter 5, experimental results of the methods introduced in Chapters 3 and 4 are presented. Finally, in Chapter 6, conclusion remarks and future work are discussed.

Chapter 2

Related Work

In this Chapter the state-of-the-art of human motion analysis, with a special emphasis on human motion tracking, human activity recognition and motion analysis of human body parts is presented. The main purpose is to show what types of analysis can be performed to study and to analyze human motion. The act of watching people to understand their actions has many potential applications in many different fields such as video surveillance, action and gesture recognition, behaviour understanding and human-computer interaction, to name a few. These topics have provided a large literature [29, 30, 31] and it is presented in the following Sections.

In Sec. 2.1, a general definition to the concept of *Human Motion Analysis* and an explanation on how tracking algorithms work are given. In Sec. 2.2 the most common method in literature for *Human Activity Recognition*, the Bag Of Visual Words (BoVW), is illustred. Issues and possible solutions of the BoVW approach are also discussed. Later in the section, the state-of-the-art of *Human Pose Estimation* and *Human Behavior Characterization* are also summarized. Finally, in Sec. 2.3, a brief description of PTZ and 360° cameras is given.

2.1 Human Motion Analysis

There is a large body of literature on human motion analysis, covering a wide range of different scientific disciplines, from natural sciences to humanities.

Human motion analysis started in Computer Vision in the early 1970s when

Neurobiology and Computational Neuroscience were motivated to acquire a deeper understanding of visual processes in humans and non-human primates [32, 33, 34, 35, 36]. This understanding led to important achievements in the computational perception theories and today its application can be found in different areas, such as surveillance, automatic video indexing, human computer interfaces, film industry, automotive, health care, etc.

Human motion analysis is a broad concept and it consists in analyzing the motion of any part of human body. An important aspect to clarify is that in human motion analysis the interest is only in the configuration of the body parts over time and not in the interpretation of their movements. The interpretation of the movement generally concerns the pose and the action recognition process, which are considered as other research topics [37].

In its simplest form, human motion analysis can be seen as the process of *detecting motion* in an image sequence, by finding the points where something is moving [32] or, as something more complex, like the process of tracking a specific object in an image sequence over time by grouping points that belong to the same object that is moving in the scene.

Pedestrian tracking, for example, does not concern with locating of individual body parts, but in general the entire body is considered as a whole and is tracked as a single object. This activity is specifically defined as *human tracking* [37].

2.1.1 Tracking of Human Motion

If we look at an arbitrary moving object, we notice that it suddenly changes its motion pattern or appearance features like shape or color. Moreover, it is common that in the environment there are different moving entities or objects that interact with each other. Keeping track of an object in a scenario like this would, in most cases, be a challenging task even for the human eye. The process of observing humans is a spatio-temporal phenomenon that inevitably involves in complex and cluttered environments, background motion, occlusions and, in many cases, also camera motion and viewpoint changes.

Since the first studies on the human visual system, many mathematical models have been developed in the field of human motion analysis to overcome all the aforementioned challenges. Just to outline an historical excursus, human motion tracking started with the optical flow and motion field methods [38, 39, 40], to recover the motion image at each pixel from the spatio-temporal image brightness variations. Since that time, many fully automatic and robust systems have been developed but human motion analysis still remained an active and a challenging topic of research [8].

Tracking is the technique used to generate the trajectory of an object over time by finding its position in every single frame of a video [41]. The goal of tracking is to segment a region of interest from a video and keep track of its motion, position and occlusion over time. In other words, a tracker assigns consistent labels to the tracked objects through video frames.

Typically, the first step in every tracking method is known as *target initialization* and it is the process of providing the initial target position in the first frame of a video sequence. In this step, a tracker identifies a region of interest to estimate its position in all future frames. The target initialization can be done by identifying the objects that are moving in the scene by means of some algorithms such as optical flow, background subtraction or frame differencing [42, 9, 43] or by means of detectors. A detector is an algorithm that deals with searching a specific object or class in a scene [44]. The use of a detector algorithm is always desirable but in real application it cannot be applied without the adoption of additional strategies, due to the fact that video objects usually suffer from significant variations in color, shape and texture over time. To allow the tracker to work properly after the initialization step, additional elements such as *appearance model*, *motion model* and a *matching and state update algorithm* are needed [6].

2.1.2 Visual Features for Human Tracking

Selecting appropriate features to correctly describe and identify the appearance of a target is a relevant process to deal with in the tracking activity. To pursue an object of interest through a long sequence of frames the target needs to be represented in a way that makes it distinguishable from the other elements present in the scene. Properties regarding the appearance and the shape of an object are usually used as a basis for its representation. Appearance or shape representations can be used alone or in combination. Based on the object visual features, the tracker should be able to reliably detect the target and guarantee a stable tracking over time [45]. For this reason, the most desirable property of a *visual feature* is its uniqueness, which enables the target to be easily distinguished in the so called *feature space* [46].

In literature, appearance based techniques are commonly divided into two categories: *image features* and *feature models*. The *image features* category groups together the image processing methods to extract features of the target. In this category the objects are usually identified by their shapes and the common ways to use this representation includes the extraction of points, edges, blobs, geometric shapes, silhouettes, active contours, articulated shape models, and skeletal models [6].

In [47] authors adopted a point-based tracking algorithm to model the motion of the target centroid in order to follow it over time. In [48], a set of low-level features like interest points, edges, homogeneous and textured regions were used to characterize a target and to allow robust tracking.

When simple rigid objects need to be tracked, primitive geometric shapes are usually adopted. In this case, objects are represented by rectangles or ellipses [49] and their motion is modeled by translation or by affine or projective transformations. Tracking algorithms based on blobs is another important group of algorithms and they were widely used in tracking applications due to their simplicity and low computational cost. Background differencing is the technique used to segment an image and estimate the blob to track.

In [50], authors proposed a technique that focused on estimating blob contours and filling their interior with a label that depends on neighboring pixel areas. Xu and Ahuja [51] proposed a contour based object tracking algorithm to track object contours in video sequences. They segmented the active contour by using the graph-cut image segmentation method. The resulting contour of the previous frame is taken as initialization in each frame. New object contour is found out with the help of intensity information of current frame and differences of current and previous frame [52].

The second category of appearance based techniques, the *feature models*, groups and models the visual features extracted by the *image features* techniques. Common appearance representations within this category are *Probability densities of object appearance*, *Templates*, *Active appearance models* and *Multi-view appearance models*.

The probability density of the target appearance [53, 54, 55] is computed in the image region in which there is the object to track. The model can be parametric, such as Gaussian [56] or a mixture of Gaussian [57], or non parametric such as Parzen windows [58] and histograms [49].

Templates based appearance representations are generally focus on simple geometric shapes or silhouettes [59] and have the advantages to carry both spatial and appearance information. The limit of this technique is that it is only suitable for tracking objects whose poses do not vary considerably between frames, due to the fact that templates encode the object appearance generated from a single view.

Active appearance models are similar to contour-based representations and are generated by simultaneously modeling the object shape by a set of landmarks. For each landmark, an appearance vector is stored which is in the form of color, texture, or gradient magnitude [60]. These type of models require a training phase where both shape and appearance are learned from a set of samples (e.g by means of principal component analysis).

Multi-view appearance models are used to encode different views of an object to allow the tracker recognizing the target even if it has suffered deformations or changes in color and texture. One way to represent different object views is to generate a subspace from given views, for instance by using PCA and Independent Component Analysis (ICA) [61, 62]. Another approach is to learn different views of an object by using statistical approaches able to learn visual representation from annotated exemplars. Machine learning algorithms such as support vector machines (SVM) classifiers [63] and Bayesian networks [64] can be used to the purpose. Recently, convolutional neural networks (ConvNets) [65] are used to compute pairwise affinity measure which determines whether two detections depict the same target or not. A clear example is given by Siamese networks [66]. In [67], authors introduced GOTURN, the first neural network tracker that learns to track generic objects at 100 fps.

2.1.3 Motion Model for Human Tracking

One of the key components of a good tracker is its ability to model the motion or the dynamic behavior of the target.

Usually, tracking applications have to run in real-time and the area where information is taken has to be limited in the next frame. The goal of using motion models consist in the prediction of the potential target position in the future frames and in the reduction of the searching area.

The motion model can be characterized by hand tuned parameters or can be learned from training sequences [68]. In literature, two main classes of motion models are mentioned: linear and non-linear. The linear motion models assume a constant velocity (CV) or a constant acceleration (CA). Their major advantage is the linearity of the state transition equation which allows an optimal propagation of the state probability distribution (e.g smoothness of velocity is modeled by enforcing the velocity values of an object in successive frames to change smoothly).

Some of the earlier motion model-based methods tried to understand the motion pattern of the target and to predict it. However, the problem with such approaches is that they can't correctly predict the abrupt motion and direction changes of the target. Examples of such techniques are Kanade-Lucas-Tomashi (KLT) feature tracker [69, 70], mean shift tracking [71, 72] and Kalman filter [73].

The state-of-the-art of visual object tracking was based for a long time on Gaussian state-space models among which the best known approach is the Kalman filter [73]. This filter was developed under very restrictive hypotheses such as the assumption that the state vector probability density function and system noise are Gaussian.

Real application trackers are usually involved in the modeling of non-linear Gaussian systems. The linear motion models soon proved to be inefficient to the non-static scenes in which non-linearity and multi-modality are significant [74, 75].

Non-linear motion models are expensive to compute and generally not adequate for real-time requirements. An example could be seen in the work [76], where a non-linear motion model was employed to track people who are free to move within a scenario. To adapt the Kalman filter to non-linearity and to non-Gaussian measurement noise, the Extended or Unscented Kalman Filters were proposed

[77, 78, 79]. Extended Kalman Filter makes the non-linear function of a dynamic system into a linear function by the use of Taylor Series. It obtains the linear approximation of a non-linear system [77].

Another important family of methods able to handle non-linear motion models is the one regarding Monte Carlo methods. Models based on Monte Carlo sampling became popular in tracking after the introduction of the particle filter method, also known as Sequential Monte Carlo, [74, 80, 81, 82]. Sequential Monte Carlo techniques for filtering time-series [83] and in particular those applied in the visual tracking field, such as [68] have quickly became famous for their ability to manage the non-linear and non-Gaussian systems in a natural way; by providing more robustness than the one offered by the Kalman filter. Sequential Monte Carlo methods combine the Monte Carlo technique with the Bayesian inference. The method implements a recursive Bayesian filter by Monte Carlo sampling. This idea was formally developed by Gordon et al. in [84], and it gave birth to many Particle Filter variants.

For its intrinsic capability to adapt to changes and to track multiple hypotheses, the particle filter algorithm was soon used for the visual tracking problem by Isard and Blake in [68, 85]. These works have shown that the particle filter framework provides a robust tracking framework which led the visual tracking problem to be reviewed as an inference problem under the Bayesian framework. In this approach, given the state-space and observation models, the posterior probability density of the target state is recursively estimated by the use of the prior probability and the available observations [86, 87, 88]. The framework estimates probabilities of predicted states based on the observed data, by sampling from a well-known probability density function.

The traditional approaches mentioned above, have typically imposed expert knowledge about the motion of the targets in their systems. Nowadays deep learning approaches have shown good performance in the learning of human motion models. Milan et al. [89] proposed a RNN-based network to learn complex motion models under the framework of the Bayesian filter. The temporal dynamics of the targets learned by the RNN are utilized to perform the state prediction and update as well as for the track management. In [90], a LSTM network is used to model and predict similar motion patterns by considering the past movements of an object and predicting its future trajectory. In [91], the authors proposed two architectures: LSTM-3LR (3 layers of Long Short-Term Memory cells) and ERD (Encoder-Recurrent-Decoder) both based on the concatenation of LSTM units. The authors noted that during inference, the network is prone to accumulate errors, and quickly produces unrealistic human motion. To overcome this problem they proposed to gradually add noise to the input during training, to force the network to be more robust to prediction errors. This solution made the network able to generate plausible motion for longer time horizons, especially on cyclic walking sequences.

2.1.4 Tracking Approaches

There is an extensive literature on tracking techniques as presented in [6] and [92]. A detailed review on the different types of tracking techniques goes far beyond the scope of this thesis, but to just give a brief overview it can be said that many different tracking paradigms exists and that tracking approaches rely mainly on a specific paradigm on the basis of their particular application. Furthermore, according to the number of tracking targets, tracking techniques can be divided into two main categories: single-object tracking (SOT) and multi-target tracking (MTT).

The difference between the the SOT and MTT techniques lies in the use of their state-space modeling. The SOT techniques deals with the modeling of the state of only one target, and all detections of other targets are assumed to be outliers. Instead, in MTT techniques there is the necessity to simultaneously take into account more than one target in the association process [93].

Single-Object Tracking (SOT)

A concise summary of techniques for SOT is now provided. The most common tracking approach based on the SOT paradigm concerns to adapt the appearance model of the target through time. Such method has the potential to allow following the target through frames at the risk of adapting its appearance to other distracting objects or to the background. Representative methods of this tracking technique are tracking of local image features [94] and model-based approaches [95]. The color histogram of the object region [71] can also be used to keep track of the appearance properties of the object.

In [96] an appearance-based method for person re-identification condenses a set of frames of the same individual into a highly informative signature, called Histogram Plus Epitome (HPE). HPE is based on a collection of global and local features. The descriptor embeds information from multiple images per person, showing that the presence of several occurrences of an individual is very informative for re-identification.

Model based tracking approaches rely on the availability of a pre-trained or an online trained detector of the selected target. These models are usually constructed off-line with manual measurement, with Computer Vision techniques or online by means of machine learning techniques. Typical representatives of this approaches are the tracking-by-detection methods such as [97, 98], or classical template matching tracking [99, 100].

In [101], a template update method for visual tracking was developed. It employs an active appearance model to account for image variation. In this method, instead of using a fixed template, the object appearance is modeled by a linear combination of appearance images. In this way, the tracking problem is formulated as a gradient descent search of the linear template combination that minimize the difference between the target object and the current appearance model.

Multi-Target Tracking (MTT)

MTT can be considered as the natural extension of the SOT paradigm to multiple targets. Although both of them share similarities, such as the suffering for occlusions, illumination variations, identity switches, camera distortions, rotations and scale changes due to the motion of the objects and of the camera, the MTT is a more complicated and challenging problem.

MTT is useful in many real applications such as autonomous driving, where the knowing of the surrounding objects can help to predict their behavior [102, 103].

In the SOT problem the interest is to design sophisticated appearance models and/or motion models to deal with challenging factors such as scale changes, illumination variations, occlusions etc. In the MTT problem, instead, there are two additionally tasks to be addressed: the first is to determinate the number of the objects in the scene, which typically varies over time, and the second is the maintenance of their identities. An additional problem is also to decide when to start new tracks (while exchanging false positives) and when to terminate tracks (while accounting for false negatives or occlusions of the objects). New tracks could be started from unassigned pedestrian detections and could be stopped if were tracked for a short time. This usually happens when a false detection shows up for a few frames and a track was initiated for it. The problem of the maintenance of the object identities is better known as *data association problem*. Since different targets move in different directions by changing their speed and their appearance over time, the development of reliable likelihood ratios are needful to address the data association problem. The data association problem is closely related to the re-identification (Re-ID) problem. To address the Re-ID problem, the existing approaches are focused on the extraction or the learning of discriminative features of the target. Usually a pair of images is given to a learner to decide whether two images come from the same person or not. Discriminative models like SVM and boosting are widely used for feature learning.

In [104], body part maps for person re-identification were introduced to solve the multi-people tracking problem. The authors used part maps to augment appearances as another feature, rather than to generate part-aligned representations.

Markov chain Monte Carlo data association (MCMCDA) was introduced for solving data association problems arising in multi-target tracking in a cluttered environment. The MCMCDA approximates the optimal Bayesian filter and operates with no or incomplete classification information, making it suitable for sensor networks. In [105], authors developed a real-time multi-target tracking that tracks an unknown number of targets by solving data association problems in a cluttered environment. In [106], authors presented a data driven MCMC method to estimate target trajectories using a batch of observations. They proposed a framework for tracking multiple targets where the input is a set of candidate regions in each frame, as obtained from a state-of-the-art background segmentation module, and the goal is to recover trajectories of targets over time.

Another application of the MTT paradigm concerns camera networks. MTT methods can be applied in a multi cameras setup but with additional issues to address, such as the searching of a stable matching between targets of two or more camera views. However, this topic is beyond the scope of this thesis.

Tracking-by-detection approach

Tracking-by-detection (tbd) has been employed in various domains [107, 108, 109, 110] and it is one of the most common used approach both for MOT and SOT [111, 112, 113]. It is widely built upon an object detector, which provides detection hypotheses to drive the tracking procedure.

In tracking-by-detection, the general idea is to first locate in each frame all objects by using an object detector and then associate detected objects to targets using features such as location in 2D frames and appearance. Tbd can be used to track generic objects by choosing appropriate detection techniques, such as [114, 115, 5] or to track pedestrians by means of numerous pedestrian detectors [55, 116, 117]. Many works also follow the conventional approach that includes the extraction of hand-crafted features [6, 7] but they are usually unable to adapt to the complex, highly non-linear and time-varying variations of the target appearance.

The work in [71] proposes a real-time color-based tracker for non-rigid objects in which mean-shift computes the most probable target position in the current frame. In [111], an ensemble of weak classifiers is trained online to distinguish between target vs. background. AdaBoost is then used to train a strong classifier by an ensemble of weak classifiers. Appearance models are widely used in tracking [118], and the challenge is making them invariant to changes of shape and lighting. In [119], the authors adopt a three-dimensional background-weighted histogram in the Hue, Saturation, Value (HSV) color space as appearance model. Hence, the mean-shift algorithm is used to search the target in the next frame. According to a simple mechanism, they modify the pan and tilt parameters to keep the target always at the center of a PTZ camera view. In [120], an adaptive background technique is used to detect moving objects; tracking is performed by histogram intersection of the color distributions of the detected foreground objects.

In the last years, tracking-by-detection methods benefited of remarkable achievements achieved by deep learning in the object detection field. In fact, newer approaches like ConvNets allowed to learn features directly from image data.

R-CNNs is a particularly successful family of methods [121, 122, 115] based on a two-stage approach in which the first stage proposes a sparse set of region proposals (RoIs), and the second stage performs object classification and bounding box regression.

Since the work of [122], the region based paradigm of the R-CNNs was adopted by many others works [123, 121, 115, 124] becoming the state-of-the-art in object detection. Region-wise features can be rapidly extracted from shared feature maps by a RoI pooling layer [124, 123]. The Faster R-CNN of [115] achieved further speeds-up by introducing a Region Proposal Network (RPN).

Alongside the two-stage networks, the one-stage network architectures have also become popular mostly due to their computational efficiency [125]. One-stage networks are usually extremely fast but their accuracy values are typically below that of two-stage networks. Recently, some multi-stage networks have also been designed and proposed. In [126] the *iterative bounding box regression* procedure is introduced, where a R-CNN is applied iteratively, to produce better bounding boxes.

Mask R-CNN [127] is built upon the Faster R-CNN [115] with two major contributions: replacing the ROI Pooling module with a more accurate ROI Align module, and inserting an additional branch out of the ROI Align module. Furthermore, Mask R-CNN is able to provide the segmentation of its detection, by adding a branch to generate masks to the existing architecture of the Faster R-CNN. The mask branch is a convolutional network that takes the positive regions selected by the ROI classifier and generates masks of size 28×28 for each region.

Recent works have also shown impressive performance in tracking by the adoption of Siamese networks. In [66], authors proposed to model the similarity between pairs of detections by CNNs. They introduced a two-stage learning scheme to match pairs of detections. First, a Siamese convolutional neural network is trained to learn descriptors encoding local spatio-temporal structures between the two input image patches, by aggregating pixel values and optical flow information. Second, a set of contextual features derived from the position and size of the compared input patches are combined with the CNN output by means of a gradient boosting classifier to generate the final matching probability.

In [128] a Fully-Convolutional Siamese Network (SiamFC) matches the candidate samples with the target template without the need of an online updating. Siamese network are supposed to learn a general matching function to tolerate the object online changes, while preserving the real-time response ability.

2.2 Human Motion Analysis for Activity Recognition

Human Activity Recognition (HAR) is a challenging task of classifying human movements that gained its popularity in the field of video analysis technology due to the increasing number of surveillance cameras and the growing demand in many multimedia applications. The automatic detection of abnormal activities to recognize dangerous behaviors (such as violent reactions, fighting, etc.) or to improve human computer interaction (HCI), as well as to help the rehabilitation of patients by healthcare systems, are just some of the many potential applications of HAR. Numerous research efforts are reported for the creation of various applications based on human activity recognition such as human gestures [129], human interaction [130], pedestrian traffic [131] and eldercare or healthcare applications [132, 133].

Regardless the field of application, the goal of human activity recognition in Computer Vision is the automatic analysis of ongoing activities from unknown videos [134] or sensors [135].

A popular review by Aggarwal and Ryoo [136] introduced a taxonomy where actions were defined as single person activities that may be composed of multiple gestures temporally organized, such as *walking*, *running*, and *punching*. In this sense the terms *action* and *activity* are interchangeable. Every human action is done for some purpose, and the ability to recognize complex human activities from videos enables the construction of several important applications in the Computer Vision field. A remarkable example is the Microsoft Kinect [137] that allows the user to interact with games by the use of gestures without any controller device.

Human actions could appear with different motion speed, appearance and pose variations. A successful action representation method should be efficient to compute, effective to characterize actions and should maximize the discrepancy between actions to minimize the classification error. Large appearance and pose variations in action categories make HAR a challenging task [136].

There are various types of human activities and these can conceptually categorize human activities into four different levels: gestures, actions, interactions and group activities [134].

Gestures are elementary movements of human body parts, and are the atomic components describing the meaningful motion of a person. Actions are single person activities and may be composed of multiple gestures organized temporally. Interactions are human activities involving two or more persons and/or objects. Group activities are activities performed by groups of persons and/or objects.

In literature, different methods exist to address the HAR problem. These methods are divided into two main categories: methods using hand-crafted motion features and methods using learning-based methods [10].

2.2.1 Hand-Crafted Motion Features for Activity Recognition

Traditional approaches for HAR are based on hand-crafted action representations. There are several ways to extract visual features (static image features and temporal visual features) and to use them to perform the recognition task.

The goal of this type of approaches is to convert an action video into a feature vector, to extract representative and discriminative information of human motion and to minimize the variations to improve the recognition performance. A classification step is then performed by training a generic classifier, such as a Support Vector Machine (SVM) [138, 139]. Hand-crafted action representation methods use features defined by experts unlike deep learning methods, which can automatically learn features from data. To capture motion information of human actions, appropriate features are constructed along both the spatial and temporal dimensions of videos. A video consists of 2D spatial (XY) images placed in a temporal (T) sequence, or equivalently, of a set of pixels in 3D XYT space. In other words, a video can be represented as a Spatio-Temporal Volume (STV) in which all information necessary to address the HAR task is present. 3D video volumes encode both spatial information of human pose over times, and dynamic information of human body. In this last structure, the concept of an individual frame and its features is replaced by an analogical voxel where densities and other characteristics are encapsulated in the volume space [140].

The most intuitive STV approach would use the entire 3D volume as feature or template and match unknown action videos to existing ones to obtain the action classification. However, this method is affected by noise and unnecessary background information.

In [1], authors, instead of concatenating entire images along time, authors stack foreground regions of a person (i.e. silhouettes). They presented two descriptors, the Motion Energy Image (MEI) and the Motion History Image (MHI) to encode the dynamic human motion into a single image by capturing the motion and the shape of the actions. The MEI method shows where the motion is occurring, while the MHI method shows both where and how the motion is occurring. The resulting image can then be used as a template that could be matched to stored models of known actions. An example of MEI and MHI can be seen in the Fig. 2.1. These descriptors depend on background subtraction and cannot tolerate moving cameras or dynamic backgrounds.

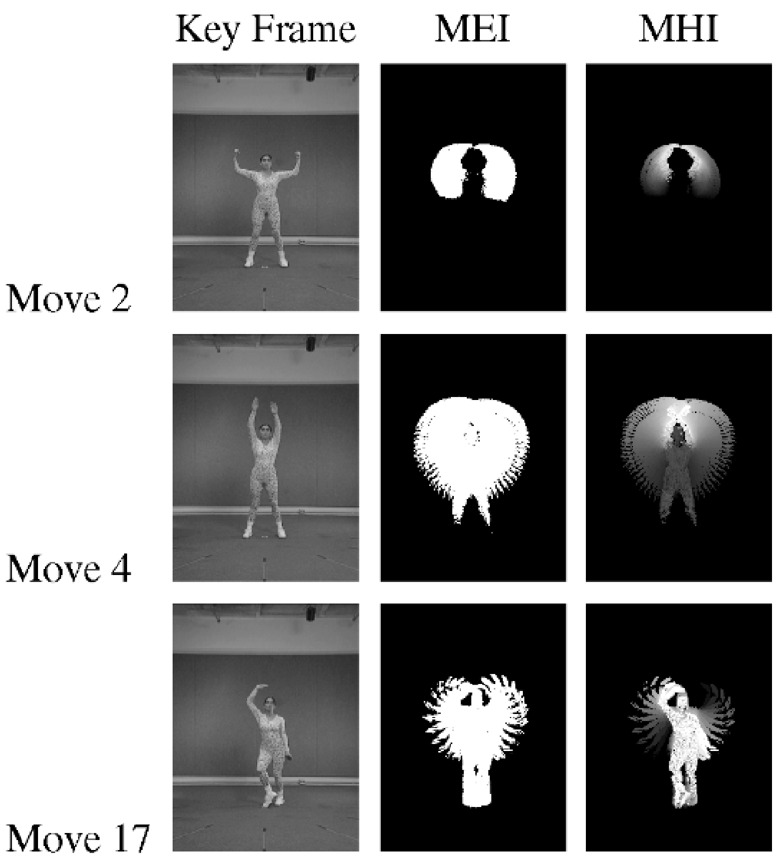


Figure 2.1: The comparison of MEI and MHI descriptors. The image is taken from [1].

An approach to compare volumes in terms of their patches has been proposed by [141]. The authors considered the spatial dimension by correlating space–time patches over different locations in space and time. To avoid the computation of optical flow, a rank-based constraint was directly used on the intensity information of the cuboids. In [142] a self-similarity descriptor that correlates local patches was proposed. The descriptor is invariant to color and texture and can deal with small spatial variations. A query template is described by an ensemble of all descriptors. Some researchers have also explored unsupervised methods for motion analysis. The authors of [143] applied a hierarchical dynamic Bayesian network model to unsupervised face expression recognition. The approach relied on previously tracked and segmented faces whose motion was described by the use of optical flow.

Hand-crafted approaches are complex to build and hard to modify. For this reason they cannot be easily adapted to new or complex data. This problem was overcome due to the rapid growth of deep learning methods for image analysis techniques. In the next section are shown various learning-based approaches, some of which still make use of hand-crafted features.

2.2.2 Learning-based Motion Features for Activity Recognition

The performance of human action recognition methods depends on the appropriate and efficient representation of data. Unlike hand-crafted action features, deep learning methods perform well with regard to automatic feature learning from images [10]. However, some of the learning-based approaches for action representation are not based on deep learning techniques. This category includes the dictionary learning based methods.

Non-Deep Learning-based Approaches

The most common approach to represent visual content in images or in videos is based on the learning of a dictionary or a visual codebook. Dictionary learning is a type of representation which is generally based on the sparse representation of input data. In dictionary-based representation approaches, the signal is represented as a linear combination of elements of a dictionary [144]. In Bag-of-Visual-Words (BoVW) approaches [24], introduced first for visual categorization in [20], visual content of images/videos is represented in terms of distributions of elements (codewords) belonging to a codebook. The BoVW pipeline is shown in Fig. 2.2. Whilst [20] adopts a Bayesian hierarchical model to learn such kind of distributions, in practice the most commonly used pipeline requires the following steps [24]: local feature extraction, learning of a codebook by means of clustering techniques (e.g., k-means), vector quantization (for discretization of the analyzed signal in terms of codewords), codewords-based histogram computation.

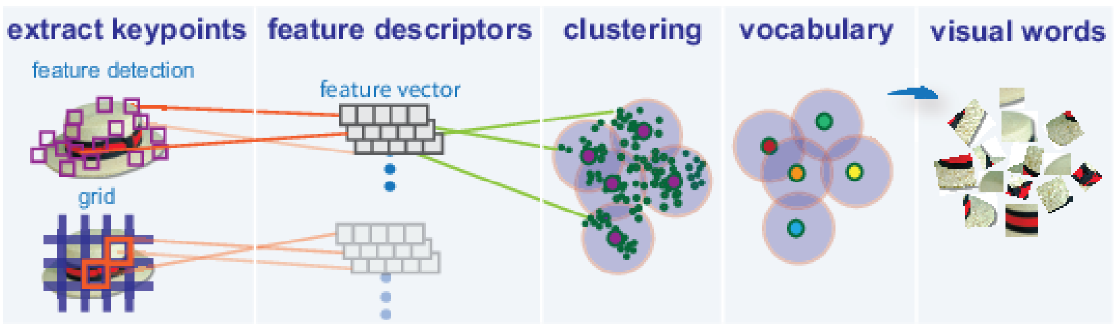


Figure 2.2: The BoVW pipeline to create a visual vocabulary, or bag of features, by extracting feature descriptors from representative images of each category. Image taken from *mathworks.com*

Such kind of paradigm has been adopted for action representation in several former works [21, 22, 145, 28, 27, 23, 25, 26] In particular, in [145], sequences are represented as a distribution of local temporal texture descriptors estimated at different time scales. A codebook of multi-scale local representations is learned via k-means, and classification is performed via SVM. In [146], a codebook of temporal windows is learned via spectral clustering of data subsequences. Similarly to [145, 146], in Ch.4 a work in which an action is represented as a distribution of temporal windows of different lengths is shown. In this work a data mining technique is adopted to learn a codebook rather than a clustering technique.

In the context of 3D Action Representation from skeletal data [11], the work in [147] represented actions in terms of co-occurring spatial and/or temporal configurations (poses) of specific body parts. A bag-of-words approach is adopted to represent an action where the codebook comprises co-occurring body-parts and is learned by contrast mining technique. In this sense, the codebook represents emerging patterns, that is patterns whose supports change significantly from one class to another. The work in [148] applied the apriori algorithm to find discriminative actionlet. An actionlet is defined as a subset of joints in the skeleton, and an action is represented by a linear combination of actionlets whose weights are learned via a multiple kernel learning approach. The work in [149] is focused on detecting reduplications in a video of American Sign Language (ASL). The method detects frequent sequential patterns of increasing length by combining smaller frequent sequential patterns, and relies on approximate matching of the discovered sequential patterns with data. In counting frequencies of patterns, a waiting mechanism is used to account for poor matching arising in presence of small misalignments between patterns and data sequence. In this sense, [149] finds gapped sequential patterns.

Another line of work considers the temporal evolution of body poses or of the appearance model, by using sequential state models such as hidden Markov models (HMMs) [132, 150] and conditional random fields (CRFs) [151].

The work in [132] addressed the problem of learning and recognizing human activities of daily living (ADL) in a room, and introduced the switching hidden semi-markov model (S-HSMM). S-HSMM is a two-layered extension of the hidden semi-Markov model (HSMM) for the recognition of human activities. The activities are modeled in the S-HSMM in two ways: the bottom layer represents atomic activities and their duration using HSMMs; the top layer represents a sequence of high-level activities where each high-level activity is made of a sequence of atomic activities.

In [151], authors relaxed the HMM assumption of conditional independence of observations given the actions and proposed to use a conditional random fields (CRF) where the state dependency is first-order. CRFs are discriminative Markov models which can use non-independent features and observations over time (contrary to the HMM assumption).

The authors of [150] proposed a max-margin method for modeling the temporal structure of the actions in a video. They use a HMM model to capture the transitions of the appearance and of the duration of actions.

In [152], authors detected space-time interest points (STIPs) with the Harris-3D detector [153], and assigned labels of -1 or 1 indicating if it belongs to the class of the interest action by using a Bayesian classifier. The feature vectors of interest point descriptors and labels are then provided to a PCA-SVM classifier to recognize the action type.

Deep Learning-based Approaches

Deep learning has the ability to process images or videos in their raw forms and automate the process of feature extraction, representation, and classification. Deep learning models can learn a hierarchy of features by building high-level features from low-level ones [154, 155, 156, 157].

Methods such as Convolutional Neural Networks [158] and Recurrent Neural Networks (RNN) [159] have shown excellent performances in many application fields [160, 161] and even outperform state-of-the-art results by automatically learning features from images, video or raw sensor data.

One of the first hybrid works using CNNs or HAR was presented by Zeng in [162] where a CNN model is used with accelerometer data. Each axis of the accelerometer data is fed into separate convolutional layers, pooling layers and then concatenated before being fed into hidden fully connected layers. This work provided a good template for how a CNN may be applied for the HAR problem, or for time-series classification in general.

ConvNets were initially conceived to recognize objects from still images [158] but had some limitations in modeling temporal information. Videos always contain two main sources of information: appearance and motion. Therefore, a spatial and a temporal stream had to be considered in neural networks to capture the appearance and the motion of an action.

This problem was investigated in [163] in which authors proposed a two-stream ConvNet architecture equipped with a spatial and a temporal network to capture the complementary information on appearance from still frames and motion between frames. This novel two-stream ConvNet architecture achieved superior results to one of the best hand-crafted based action recognition method [12].

Even 3D ConvNets [164] were designed to overcome the limits of handling 2D inputs. In this work, the authors extracted features from both the spatial and the temporal dimensions by performing 3D convolutions, thereby capturing motion information encoded in multiple adjacent frames. The downside of this type of networks is that they are not suitable for real-time applications due to their computational complexity.

In the work of [165] it was proposed another 3D ConvNet for human action recognition which was evaluated on four publicly available datasets, confirming not only that 3D ConvNets are more suited for spatio-temporal features than 2D ConvNets but also that ConvNets architecture with small $3\times3\times3$ kernels are the best choice for spatio-temporal features.

Traditional methods modeled the spatial structure and temporal dynamics of human skeleton with hand-crafted features and recognized human actions by means of well-designed classifiers. Instead, in [166] the authors proposed a RNN for skeleton-based recognition. The human skeleton was divided into five parts according to human physical structure and then separately fed into five subnets. The representations extracted by the subnets are then hierarchically fused to be the inputs of higher layers and the final representation was fed into the output layer.

Some studies combined hand-crafted features into ConvNet models to improve the performance of the action recognition task. An example of data fusion method for the action recognition task was proposed in [167] where the feature fusion of two networks is accomplished through element-wise multiplication with bias. Data fusion allows to a network to learn a combined representation of multiple input streams.

Another important challenge to consider in the HAR is the view variance of the human action. The same action viewed from different angles looks quite different. To address this problem the authors of [168] proposed a human pose representation model that transfers human poses acquired from different unknown views to a view-invariant high-dimension space. The model is a deep ConvNet able to generate training data by fitting synthetic 3D human model to real motion and by rendering human poses from numerous viewpoints.

The confusion between different activities that express similar poses (e.g., running and walking) are common issues to asses in the HAR task. One action category may contain multiple different styles of the same human movements. The solution of this problem is to increase the inter-class variations between the various actions. Accurate and distinctive features need to be designed and extracted from activity videos to deal with these problems.

Long Short Term Memory (LSTM) network [169] has the property of being invariant to pattern shifts in the time domain [170] and is considered a natural model for time-series.

In [171], authors proposed an end-to-end fully connected deep LSTM network for skeleton based action recognition inspired by the observation that the co-occurrences of the joints intrinsically characterize human actions. Taking the skeleton as the input at each time slot and introducing a novel regularization scheme, co-occurrence features of skeleton joints are learned in this way.

In [172] a gesture recognition system employed a shallow bidirectional LSTM to a problem of 14-class gesture classification with only one forward hidden layer and one backward hidden layer to explore long-range temporal dependencies.

The authors of [173] inspired by the graphical structure of the human skeleton proposed a spatio-temporal LSTM model for 3D human action recognition, which extends RNNs to spatio-temporal domains to analyze the hidden sources of action-related information.

2.2.3 Human Pose Estimation

In Computer Vision, humans are typically considered as articulated objects consisting of rigidly moving parts connected to each other at certain articulation points known as human joints, keypoints, elbows and wrists. Human Pose Estimation (HPE) is defined as the problem of locating these human joints and inferring 2D or 3D human body part positions from still images or videos. HPE methods aim to capture human poses to construct a human skeleton on the basis of the captured body joints. In these terms, a human pose skeleton represents a set of coordinates that can be connected to describe the pose of the person in a graphical format. Skeleton poses do not suffer the problem of the action intra-classes variances in the same way of the classical appearance-based approaches [174].

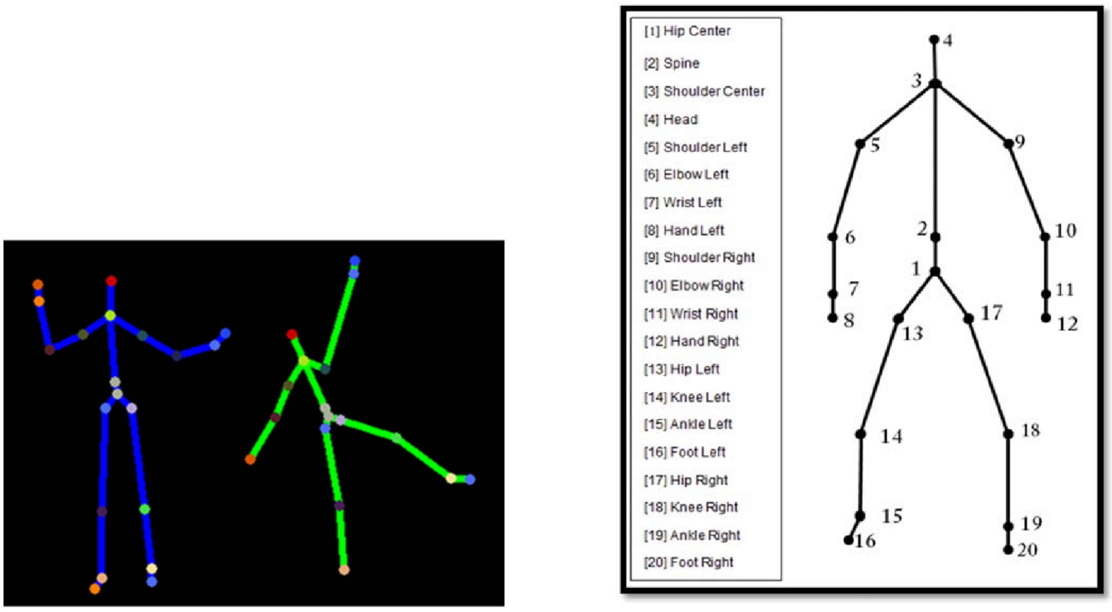


Figure 2.3: On the left, 3D skeleton joints tracked by the Microsoft Kinect (v1) are shown. On the right, the graphical representation of the 20 joints that composes the skeleton model used by Kinect is presented. Images are taken from [2].

A sample human skeleton is shown in Fig. 2.3.

In literature, there is a wide variety of works that efficiently estimate human body poses by means of the extraction of human skeletons [175, 140, 176].

Action recognition and human pose estimation are closely related but both problems were generally handled as distinct tasks. In [177], authors demonstrated the superiority of the use of features based on pose estimation in the context of an action recognition scenario. They evaluate an action recognition task in a home monitoring scenario by using the same classifier and same dataset but different features. They compared appearance based features, pose based features and a combined approach of the previous two for action recognition purposes. Visual features were colour, dense optical flow and spatio-temporal gradients. Pose based features were qualitative geometric features [178]. Their results showed that the optimum approach was the one based on pose features that significantly outperformed the appearance-based approach and was even slightly better than the combined approach.

In [179], authors detected key-frame poses in walking sequences and initialize a local appearance model to detect body parts at intermediate frames.

Nowadays, approaches for human pose estimation are usually divided into hand-crafted and deep learning methods.

The performance of the state-of-the-art of the classical human pose estimation methods was improved by the use of the deep learning approaches.

OpenPose [180] is one of the most popular methods for multi-person human pose estimation. The network first detects keypoints belonging to every person in the image, followed by assigning parts to distinct individuals. The architecture of the OpenPose network is showed in Fig. 2.4.

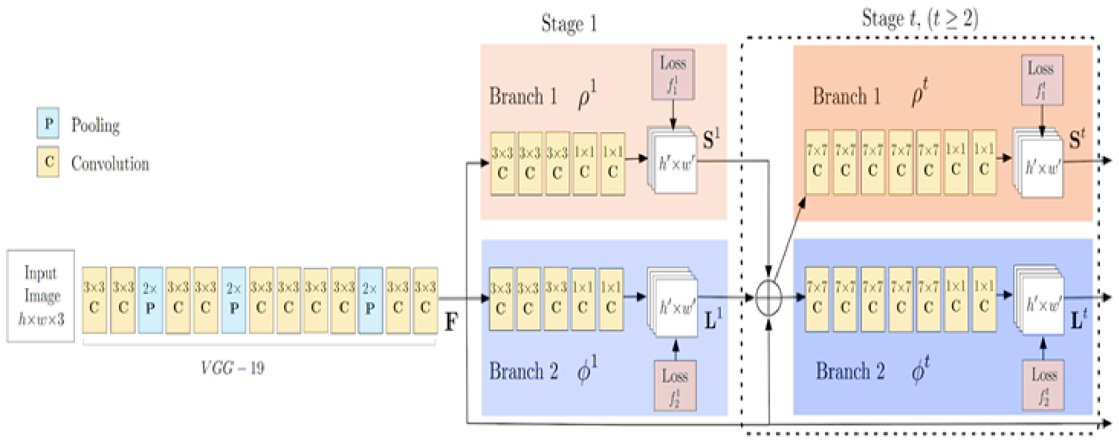


Figure 2.4: Flowchart of the OpenPose architecture.

The network extracts features from an image by using the first few layers of a VGG-19. The features are, then, fed into two parallel branches of convolutional layers. The first branch predicts a set of 18 confidence maps, with each map representing a particular part of the human pose skeleton. The second branch predicts a set of 38 Part Affinity Fields (PAFs) which represents the degree of association between parts. Successive stages are used to refine the predictions made by each branch.

DeepCut [181] is another method for multi-person human pose estimation based on integer linear programming (ILP) that jointly estimates poses of all people present in an image by minimizing a joint objective. This objective aims to jointly partition and label an initial pool of body part candidates into a consistent sets of body part configurations, each corresponding to distinct people.

2.2.4 Human Behavior Characterization

In their most general definition, behaviors are defined as "the internally coordinated responses to internal and/or external stimuli" [182]. In humans, this translates to the individuals' responses related to internal or perceived environmental stimuli, mediated by psychological states.

The study of such responses is carried out by psychology, psychiatry, neurosciences, and, more in general, by cognitive sciences, with the goal of providing clues about the inner mechanisms of the human brain that underline perceptual and decisional processes [183]. While this knowledge can facilitate people personal and interpersonal functioning [184], those studies become particularly useful in the treatment of complex psychopathologies where such mechanisms are damaged [185]. Notably, such insights on human behaviors can also be applied to the design of products, systems and devices we use every day by making them easier, more comfortable and less frustrating [186].

Computer science contributed to such studies not only as a metaphor for innovative computational models [187] able to describe cognitive processes but also with useful methods and tools to capture and automatically or semi-automatically characterize humans' behaviors [188, 189]. However, automatic/semi-automatic behavioral analysis is not straightforward since it is unclear how observable behaviors should be measured and characterized. The problem is made more difficult given the great behavioral heterogeneity that can characterize humans' responses, in particular during social interaction in which individuals' behaviors arise from interpersonal exchanges. Moreover, factors such as age, motor capabilities or the presence of cognitive impairments can make the problem even harder.

Studies about observable human behaviors take advantage of measurements of multi-modal characteristics [18] such as gait, body poses, body movements, eye gaze, facial expressions speech or behavioral time talking. All these measurements are derived from signals acquired by a wide variety of sensors (microphones, cameras, wearable sensors such as accelerometers or gyroscopes) and led to represent human behavioral responses in terms of one or more synchronized, multi-modal signals [19].

In literature, the problem of characterizing humans' behaviors has been approached mainly through three methodologies:

- **Manual annotation**: trained human annotators carefully label the observations according to a previously agreed set of labeling criteria. Consistency between annotators is verified through inter-rater reliability measures like the Cohen's kappa or the Intra-class Correlation Coefficient (ICC) [190];
- **Automated annotation**: observed measurements are automatically processed according to a set of rules that can be explicitly programmed or implicitly inferred from data [191, 192];
- **Semi-automated annotation**: a subset of manually annotated measurements is employed as sample pattern into an automated system that characterizes the remaining observations accordingly [193, 194, 195].

Manual annotation fully relies on the human effort to characterise and annotate observations. While it can achieve an elevated degree of precision, it is extremely expensive in terms of temporal and human resources demanded to the human annotators. Recently, crowd-sourcing systems such s the Amazon's Mechanical Turk has been used to reduce such costs. However, it is unclear if the annotation collected with such systems would have the minimum quality required for fine-grained human behavior analysis from a psychological point of view. At the same time, ethical issues arise in the case of analysis of sensitive data [196].

Fully automated annotation systems can be effective when explicit rules are an adequate tool to achieve the requested degree of performances; as an alternative, rules can be inferred through clustering techniques able to reveal pattern and recurrences in the data. However, also in this case, a post-processing revision by an expert is needed in order to interpret and validate the found clusters.

In recent years, impressive performances have been achieved through the use of deep neural networks [197, 198]. However, such performances are associated with the exploitation of very large annotated training sets that, in the field of behavioral analysis, may not be available. Furthermore, such models are very complex and still difficult to interpret. On the contrary, semi-automated annotation tries to combine the benefits from both the automated and manual annotation methodologies while minimizing their drawbacks. First, this methodology tries to limit the human annotators' effort in labeling the data. Secondly, it attempts to develop models that are easy to check and interpret for experts that are not necessarily computer scientists. Although a manually annotated set of observations is still required, interesting performances can be obtained from a set of data that is smaller than the one required to train fully automated annotation systems. Also, since sample patterns are supplied to the system, more control and more understanding of the underlying rules can be gathered.

In automated and semi-automated systems, complex behaviors are modeled in terms of dynamical systems. In particular, their temporal dynamics can be expressed as a sequences of simple, stationary or quasi-stationary dynamical processes, each one characterised by its own set of parameters. In this sense, complex behaviors emerge as the evolution in time among such simple behavioral classes [199]. A simplified model of this complexity would be a finite state machine in which states represent simple behaviors while transitions from state to state represent changes from a behavioral class to another. Consequently, in correspondence to such transition, the parameters of the dynamical system will change in accord to the parameters of the behavioral class involved in the transition itself. For this reason, such transitions are generally known as change-points [14, 200, 201, 202, 203].

According to the level of details used to characterize the observed behavior, three main techniques can be adopted:

- **Global characterization**: the transitions among behaviors are ignored. Observed measures are treated as part of a single dynamical process characterized through comprehensive statistical descriptors [204];

- **Slicing**: as in the global characterization, transitions are ignored but behaviors are described by statistical descriptors of a finite sequence of temporal slices of the observed measures [205, 199];

- **Local characterization**: change-points are explicitly detected and exploited to segment and characterize the observed measures [206].

Global approaches achieve a rough characterization of humans' behaviors, ignoring the details of its evolution in time. Consequently, they can be very effective in revealing and describing features that are stationary among the whole considered time frame. In contrast to the global approaches, local techniques may result in a finer and detailed characterization of humans' behaviors [207, 208]. Nonetheless, due to the inner stochastic nature of the behavioral measurements, such characterization are more sensitive to small fluctuations ascribable to noise and to the performances of the sensors used to observe the behavior [209]. In this sense, by considering time-slices of the observed behaviors, slicing techniques may help to reduce or partially filter out these fluctuations. As a consequence, a statistical description of the behavior would reveal quasi-stationary features among different slices. Such methodology can be particularly useful to describe how observed behaviors evolve in time in a more refined way. Notably, this technique can be used to study first impressions and early events [210], to analyze long-term scenarios [211] or to evaluate before-and-after effects [212] of particular events [14].

On the other hand, local analysis of the observed measurements can highlight breakpoints and transitions from one behavior to another. Such change-points are a powerful tool to segment the observations and reveal events, correlations, causalities and synchronous phenomena, but also to discover abnormal behaviors or delays in the expected behavioral changes.

2.3 Cameras and Applications

The most used security cameras were simple standalone cameras that captured videos of the area in front of them. For many years, this technology remained fairly unchanged until new technological pushes contributed to their improvement. These new improvements satisfied the increasingly request of actively monitoring people and places [213]. A first step in this direction was made with the introduction of pan, tilt, zoom (PTZ) camera devices. PTZ cameras are particular devices that can be remotely controlled to direct the attention to interesting events.

2.3.1 Pan-Tilt-Zoom (PTZ) Camera

The introduction of pan-tilt-zoom (PTZ) cameras brought new capabilities as well as new problems to be solved. PTZ cameras have the ability to acquire high-resolution imagery and manually track events over a wide monitored area. An example of this type of cameras is shown in Fig. 2.5.

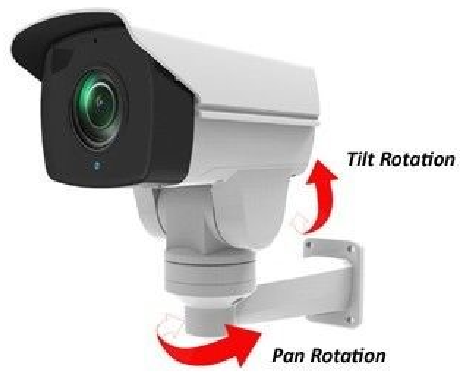


Figure 2.5: A PTZ camera is a camera with pan, tilt, and zoom functionality. These devices can be remotely controlled left and right (pan), up and down (tilt) and they can zoom in and out. The maximum range for pan and tilt angles are between -180° and +180° .

Surveillance systems with such cameras are usually based on a network of static cameras that monitors a wide area and that provides information so the PTZ camera can pan and tilt to the required position and observe or follow target movements [214]. Due to their wider field of view, the required number of static cameras in an environment can be significantly decrease.

Although there are intrinsic advantages in the use of PTZ cameras, their application is still a challenging research topic in Computer Vision. The difficulty of creating good vision-based PTZ tracking system to automatic track objects or pedestrians in different environments affected by illumination changes, occlusion and cluttered background is one of the main issue for this type of camera. In fact, due to changes in the environmental conditions (lights, shadows, pedestrian/object movements, servomotor control, etc.) the results of a PTZ tracking algorithm are difficult to reproduce.

The difficulty in evaluating PTZ tracking algorithms arises from the dynamic nature of this type of cameras. PTZ tracking algorithms have to deal with both locating the target in the image and controlling the motors of the camera to aim that the target stays at the center of its field-of-view (FOV). Since the camera control is not instantaneous and the delay required for re-positioning the camera can negatively affect object tracking, the problem becomes relevant when a target changes abruptly its velocity or get occluded and disappears from the camera view before the camera control phase ends.

Data acquired by a PTZ camera change every time the pan, tilt and zoom parameters are adjusted and for this reason tracking algorithms results are difficult to reproduce and to compare. These are just few of the reasons for the lack of efficient PTZ evaluation metrics to asses the quality of the estimated pan, tilt and zoom parameters by PTZ tracking algorithms.

2.3.2 Virtual Pan-Tilt-Zoom (vPTZ) Camera

Authors of [215] presented a simulated virtual world with animated pedestrians to allow to various virtual sensors (including virtual PTZ cameras) to track objects or pedestrian. This novel approach was very interesting as it permitted repeatable evaluation of trackers in the context of sensor networks. Its limitation is that it does not reproduce real world settings, such as change in lighting conditions, or it does not address the limits of real camera sensors (resolution, motion blur, etc.) because scenes are artificial.

Another attempt to create a virtual environment to allow the evaluation of PTZ tracking algorithms in controlled and repeatable scenarios was that of [216]. In this work, authors projected a ground-truth video on screen in front of the camera in order to calibrate the system and asses the tracking results based on the target position on the screen. The aim of this paper was to propose a kind of platform to evaluate different single-target tracking algorithms for PTZ cameras, based on the principle of the repeatability. As mentioned above, the repeatability of results with PTZ camera tracking algorithms is difficult to achieve, because these type of camera see a different scenario on the basis of the choice of the set of the pan, tilt, zoom parameters. This choice is obviously depending on the tracking algorithm and the lack of a unique video benchmark which allows genuine global testing is the main issue in the development of PTZ trackers.

Although the idea to project a ground-truth video on screen in front of the camera is effective in terms of reproducibility of tracker results, it certainly cannot be considered as a possible standard evaluation system due to the difficulty of adopting similar equipment, which can be impractical in some cases.

The introduction of 360° camera technology offered more sophisticated approaches to simulate PTZ cameras. These approaches exploit the availability of equirectangular or cylindrical images produced by 360° cameras to simulate PTZ cameras. One of the most interesting work was presented in [3]. The key idea was to use a PTZ camera simulator that permits to pan, tilt and zoom on the basis of a spherical video captured online. Authors proposed a framework which allows to maintain unchanged tracking scenarios for each experiment. According to this, it was possible to replicate online PTZ camera control and behavior including camera positioning delays, tracker processing delays, and numerical zoom.

Further ideas have been presented in [217], where authors proposed a framework to simulate an unlimited number of PTZ cameras for tracking purposes. In [15] we presented the main idea to build a virtual PTZ camera for tracking purposes similar to the one of [217]. It essentially consists of grabbing panoramic images from a 360° video to build a vPTZ. This ensures reproducibility of tracking results by keeping fixed the experimental conditions. More details on vPTZ cameras are presented in Ch.3.

2.3.3 Applications

Nowadays, the use of video cameras is no longer confined to video surveillance and many different types of cameras, initially conceived for video surveillance, are now applied in numerous other domains. Just to cite few examples, PTZ or 360° camera can be used to automatically pan, tilt and zoom on the active speaker during a video conference, webinar or virtual classroom [218, 219]. In a virtual classroom, the PTZ camera can actively be focused on the educator delivering the lecture and, if a question is asked by a student, it can automatically be moved towards the speaker. In [220], a method to track the position and the speaking activity of multiple subjects in a meeting room is presented. The tracking problem is modeled with a Markov state space in which the hidden states represent the multi-object configuration (e.g., position and scale) and the observations are given by data acquired from microphones and multiple cameras.

Another interesting application of 360° cameras is introduced in [221] where a computer-generated reconstruction of a church was adapted to create a panoramic film. The reconstruction is presented in a panoramic viewer and also on a wrap-around projection system. It is focused on the fundamental principles of creating 360° films and in how to record 360° video using panoramic cameras inside heritage sites.

In [222], a simple sketching strategy that allows simple two-dimensional panoramic sketches to be rendered in pseudo-three-dimensions using panoramic viewers is presented. It allows designers without three-dimensional modelling experience to represent their design ideas that can be explored from single points in space. In [223], authors combined 360° videos with virtual reality (VR) techniques to let users to experience the content and to interact with the environment, rather than to just watch it. These technological resources offer an immersive experience in which each person can choose where to look at. The authors used this virtual environment to present a teaching experience of 360° immersive visualization of an operating room and of an anatomical dissection room for learning and training purposes.

Chapter 3

Tracking in 360° Videos

Video tracking is the process of locating one or multiple moving objects over time by using cameras. Tracking people with cameras is a well-known problem in Computer Vision and in the literature it is known as *pedestrian tracking* or *human tracking*. Many methodologies were developed to analyzes video sequences and to retrieve the motion of one or more targets at each frame, especially with static cameras. The introduction of pan-tilt-zoom (PTZ) cameras brought new capabilities, but also new challenges. The benefits of having a PTZ camera are obvious: PTZ cameras can be remotely controlled and can cover a much larger area than the one covered by the use of many static cameras. This provides cost savings and the possibility to direct the attention to suspicious events. Unlike static cameras, which have a fixed position and orientation throughout tracking experiments, PTZ cameras adaptively alter their orientation to track targets. This is why PTZ tracking algorithms can only be performed online. The idea of creating virtual environments for PTZ cameras (vPTZ) permitted not only the comparison of PTZ tracking algorithms under the same environmental conditions but also the creation of novel PTZ tracking algorithms. The virtual environment presented in this thesis was built on the base of equirectangular images acquired by 360° cameras. Thanks to the use of the vPTZ framework we developed two pedestrian tracking algorithms for 360° cameras based on the tracking-by-detection approach and one based on the Bayesian approach. The tracking-by-detection approach is generally implemented by starting with the pedestrian location hypotheses generated by a person detector. Instead, the Bayesian approach involves an object model with an associated observation density function and a mathematical method to sequentially infer the posterior probability density. The pedestrian algorithms presented in this Chapter are single-target and are initialized in the first frame with the target to follow. Then, the trackers automatically track the target trajectory over the subsequent frames.

To summarize, in Sec. 3.1 the vPTZ framework for the development of the 360° tracking algorithms is presented. On the base of this virtual environment, in Sec. 3.2 and in Sec. 3.3 two algorithms based on the tracking-by-detection paradigm and one based on a Bayesian approach are respectively introduced. These algorithms had been tested on the same dataset [3] to obtain an effective comparison. Their results are presented and discussed later in the Ch. 5.

3.1 From Spherical Views to Virtual PTZ camera Views

The recent introduction of 360° cameras gives the possibility to acquire spherical images of the environment to monitor. Spherical images acquired by 360° cameras are obtained by stitching together images acquired from different viewpoints by several optical sensors on-board of the camera. Spherical panoramas incorporate a 180°vertical viewing angle and a 360° horizontal viewing angle. They contain light data originating from all directions, and therefore can be visualized as comprising the points on a sphere. Then, they offer a complete view of the scene in a single image and for this they can be used to reduce the number of devices needed in the tracking problem. These particular images are generally stored in specific formats, one of these is called *equirectangular*, in which the coordinates of each pixel represent latitude and longitude of the corresponding point on the sphere surface. An equirectangular panorama simply consists of a single rectangular image whose width and height are correlated as $2 : 1$. Equirectangular images are stretched in the latitude direction and it is the reason for a considerable amount of data redundancy near the poles. An example of an equirectangular image is shown in Fig. 3.1

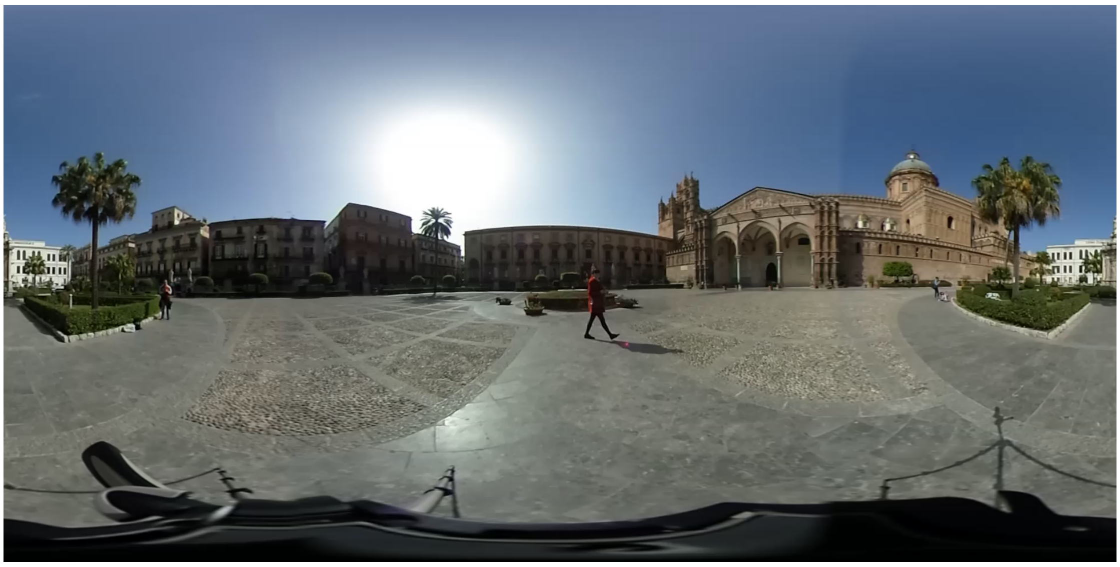


Figure 3.1: An equirectangular image of the the Cathedral in Palermo, Sicily

By using a simple geometrical transformation it is possible to map a specific portion of the equirectangular image onto a tangent plane and to get different projections of the spherical surface. Tracking a point on the spherical surface is not different from controlling a PTZ camera: latitude and longitude of a point in the sphere correspond respectively to the tilt and pan parameters of a virtual PTZ camera. The zoom parameter of a PTZ camera can also be easily simulated by controlling the extension of the spherical surface to project on the tangent plane. Such tangent plane can be interpreted as the image plane of a virtual PTZ camera oriented towards the point of tangency. This is the main idea that we used for tracking purposes to simulate an unlimited number of PTZ cameras by using simple offline 360° videos.

3.1.1 Mathematical background

360° images are stored as $2D$ projections of the captured $3D$ world on the surface of a viewing sphere. There are various popular projections to map a sphere to a plane and these are mainly used in cartography applications. Stereographic, cylindrical, Mercator and equirectangular are just some of the most common projections [224]. Each of these, as mentioned before, involves some distortions during the projection on the tangent plane. The projection that turns out to be optimal for tracking purposes is the equirectangular projection because the most noticeable distortion in this type of projection is the horizontal stretching that occurs as one approaches the poles from the equator of the sphere, this culminates in the poles (a single point) being stretched to the whole width of the map. In this way the distortion is limited only to the poles of the sphere that represent the highest part, the sky and the lowest part, the floor, of an environment. Thus, the central area of an equirectangular image is the most interesting one for computer vision analysis and with this type of projection it does not suffer from any kind of distortions.

Each pixel (x_r, y_r) in an equirectangular image is uniquely defined by two angles: the latitude $\phi \in [-\frac{\pi}{2}, +\frac{\pi}{2}]$ and longitude $\lambda \in [-\pi, +\pi]$. Conversion from pixel coordinates of the equirectangular image to latitude and longitude on the spherical surface entails a simple scaling of the pixel coordinates.

To map the spherical surface onto a tangent plane one of the possible transformation that can be used is the gnomonic one [224].

We define the sphere center O, and we consider a point $S = (\lambda_S, \phi_S)$ on the sphere surface. The gnomonic transformation allows us to map a point $P = (\lambda_P, \phi_P)$ on the sphere surface to a point P' on the plane tangent to the sphere in S [225]. The way this projection works is shown in Fig. 3.2.

The point S represents the center of the imaging plane and, hence, it is assumed to be equal to $(0, 0)$ in the image coordinate system. The projection P' has coordinates (x, y) on the image plane and can be explicitly computed by means of Eqs. 3.1 [224].

$$\begin{aligned} x &= \frac{\cos(\phi_P)\sin(\lambda_P - \lambda_S)}{\cos(\theta)} \\ y &= \frac{\cos(\phi_S)\sin(\phi_P) - \sin(\phi_S)\cos(\phi_P)\cos(\lambda_P - \lambda_S)}{\cos(\theta)} \\ \cos(\theta) &= \sin(\phi_S)\sin(\phi_P) + \cos(\phi_S)\cos(\phi_P)\cos(\lambda_P - \lambda_S) \end{aligned} \tag{3.1}$$

It is also possible to derive the inverse transformation, which is computed by means of Eqs. 3.2 [224].

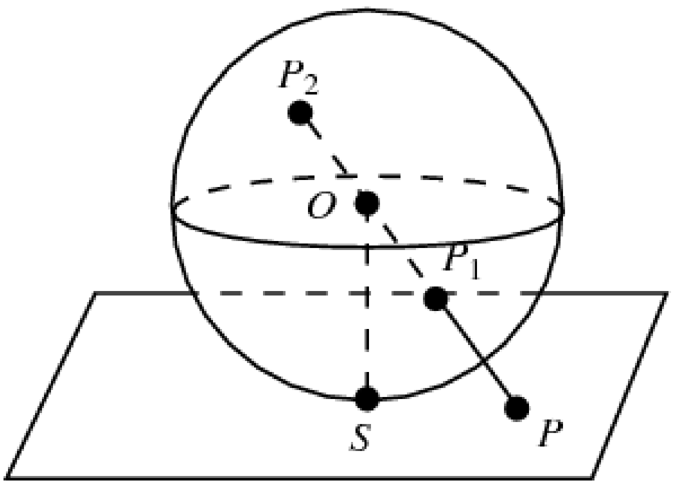


Figure 3.2: Gnomonic projection, image source: http://mathworld.wolfram.com

$$\begin{aligned}
&\phi_P = \sin^{-1}\left(\cos(\theta)\sin(\phi_S) + \frac{y\sin(\theta)\cos(\phi_S)}{\rho}\right) \\
&\lambda_P = \lambda_S + \tan^{-1}\left(\frac{x\sin(\theta)}{\rho\cos(\phi_S)\cos(\theta) - y\sin(\phi_S)\sin(\theta)}\right) \\
&\rho = \sqrt[2]{x^2 + y^2} \\
&\theta = \tan^{-1}(\rho)
\end{aligned} \tag{3.2}$$

By setting the point S, the projected image can be easily obtained by means of the inverse transformation: for each image pixel,the corresponding point on the spherical surface is computed and, by simple rescaling of latitude and longitude, the pixel coordinates on the equirectangular image are calculated. Further improvements can be applied to limit noise and artifact in the projected image, such as the bilinear interpolation of the color levels.

An example of the gnomonic projection is shown in Fig. 3.3

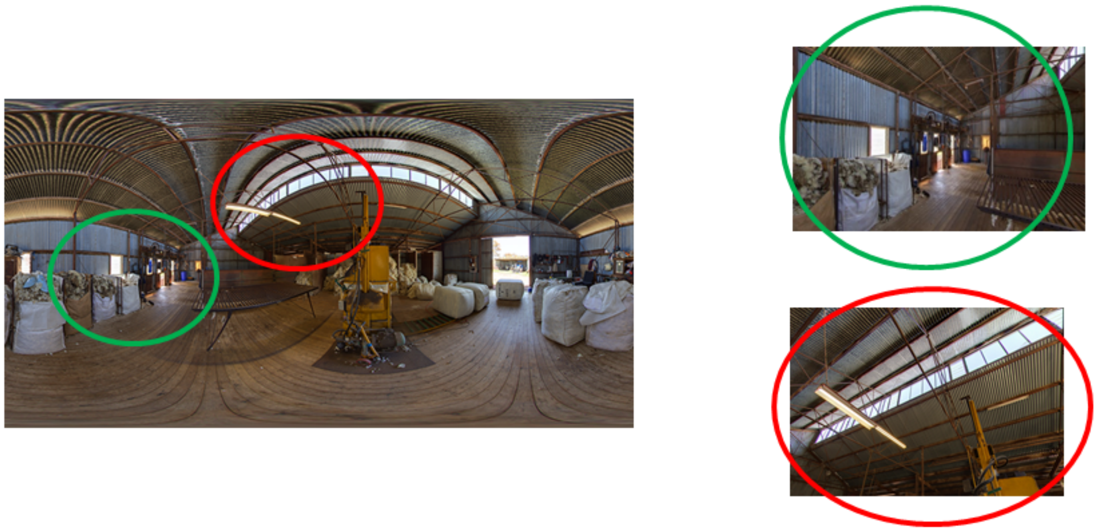


Figure 3.3: The gnomonic projection allows the mapping of a specific portion of the spherical view onto a tangent plane. By varying the tangent plane, it is possible to get different projections of the spherical surface. The red and the green circle show the projected portions of the equirectangular image.

3.1.2 Zooming in vPTZ

The zoom parameter in a PTZ camera controls the angle of view of the camera. The effect of zooming in a virtual PTZ camera can be simulated. Projection on the image plane is computed by defining a grid of points and by applying, to each point, the inverse transformation described in the former Section. The grid is calculated in such a way that the center of the image corresponds to the origin $(0, 0)$. In our framework, we calculated the grid by setting the size of the projected image to $N \times M$ and by sampling point coordinates along rows and columns in the range $[-1, 1]$ with a uniform step equals to $\frac{1}{N}$ and $\frac{1}{M}$ for the rows and the columns directions respectively.

To zoom out, it is sufficient to multiply the pixel coordinates in the grid by a zoom factor greater than 1. This has the effect of increasing the size of the spherical surface projected on the image plane. On the contrary, to zoom in, it is sufficient to multiply the coordinates of the points in the grid by a zoom factor lower than 1. This has the effect of shrinking the spherical surface that is projected on the image plane. The zoom factor is defined as $\tan\frac{\theta}{2}$ where θ is the camera field of view, which ranges in $[0, \pi]$.

One of the main challenges with PTZ camera is to automatically adjust the zoom while keeping the target well visible in the virtual camera frame. In this work, we keep the zoom fixed in order to be able to compare against [3]. However, a method to dynamically adjust the zoom could be easily added to our tracker approach. For example, it might be possible to adjust the zoom based on the ratio between the target bounding box area and the virtual camera frame area such that the size of the target remain constant in proportion to the projected image. This problem remains a topic of future investigations.

3.1.3 Tracking in 360° videos

A 360° video is a sequence of spherical images of the environment to monitor captured by an omnidirectional camera or combined by multiple cameras to cover the complete horizontal field of view. Relative to ordinary videos, 360° videos are usually stored in an equirectangular format and this does not allow the use of ordinary tracking algorithms designated for static cameras. These algorithms may not perform well on 360° videos because of their unique characteristics. In 360° videos the occlusion problem is almost unavoidable. Indeed, in this type of videos an object may disappear from the left but reappear on the right border, or disappear from the top and then reappear on the bottom. Nonrigid deformation is another obvious artifact on the equirectangular format. Furthermore, due to the continuous changes of the FOV of the 360° camera, light and scale changes occur much more frequently than in ordinary videos. Because of these problems trackers designed for static cameras tend to confuse or lose the tracking target under these circumstances in 360° videos. 360° video trackers can essentially be divided into two categories: those that trace directly into the equirectangular image and those that use some sort of projection or spherical kernels as in [226, 227], to map a section of the equirectangular image to a plane. Performing tracking directly onto equirectangular images is challenging and not very convenient because the target appearance would depend on the location of the target on the equirectangular image itself, and the image sphericity must be taken into account during tracking, with a considerable computational cost.

At the best of our knowledge, evaluation of the state-of-the-art tracking algorithms on 360° videos is not yet covered in any works. The only attempt is that of [228] in which the authors used a virtual PTZ framework to evaluate different tracking algorithms of PTZ cameras and to compare their performances. In this work the authors used the vPTZ framework presented in [3] to evaluate in a reproducible way trackers in PTZ scenarios. Their vPTZ framework is based on spherical videos captured offline and it is also capable to simulate the mechanical delay of the PT motors. They evaluated 19 different tracking algorithms and compared and analyzed their performances by means of the metrics introduced by [3]. Although the idea in [228] is innovative, the authors did not modify the tracking algorithms of the static cameras to be adapted for their vPTZ camera framework.

What they did was simply to execute the existing algorithms directly in the image plane for the current camera viewpoint (i.e. the viewed subregion on the image sphere projected on the camera image plane) without considering all the issues of the 360° videos exposed above (frequent light and scale changes, occlusion handling, nonrigid deformation, zoom handling etc ·). They only extended their framework to add target position prediction for the next frame, accounting for camera motion and processing delays.

In the state-of-the-art there are present some works that have adapted tracking algorithms for static cameras to work in 360° videos. In [229] the authors adapted the Kernelized Correlation Filter (KCF) tracking algorithm to work on 360° videos. They proposed a method based on KCF for omnidirectional vision, in which they made some improvements about scale and occlusion by combining the multi-scale KCF with Kalman estimate. In [230] the Track-Learn-Detection (TLD) tracker was modified to fit the needs of 360° videos. Their Modified Learning-Training-Detection (MTLD) imposes important modifications on the TLD components to elaborately adapt it to solve the problem of unknown object tracking in 360° images. The proposed MTLD method outperforms the TLD method significantly in the object tracking activity, in the case of high rate of out-of-plane rotation and in the handling of rapid environment changing.

Nevertheless, although there is some work on object tracking at 360° videos, there is still no evaluation of the state-of-the-art of tracking algorithms on 360° videos and there is no study of the general issues that need to be addressed in this type of videos.

The problem of 360° object tracking in images is much more complicated than rectangular tracking. Applying conventional tracking methods for object tracking directly in 360° video is very challenging due to distortion of 360° images caused by natural convexity, concavity and rotation of objects in the 360° images. Furthermore, the high resolution of 360° images limits the tracking algorithm speed necessary to maintain real-time capability. Without the capability, the tracking of objects which rapidly get far away out the scene is not possible. For these reason the conventional tracking algorithms cannot be applied directly in 360° images without any improvements. In this thesis to overcome these difficulties the main idea is to use the equirectangular image as world representation and the virtual camera plane as if it was the output of a traditional PTZ camera. The tracking model takes the equirectangular projection as input but the tracking framework is performed in the virtual camera plane. Supposing that the virtual camera parameters pan and tilt coincide with the latitude ϕ and longitude λ of the equirectangular image and that zoom coincides with the filed-of-view of the projection, the tracking method applied in the works of this thesis takes care of estimating, for each virtual frame, to update the parameters to keep the target in the center of the virtual camera plane. In the works presented in this thesis different strategies have been developed in this sense. The task of tracking can be generally described as two steps, detection of objects (such as pedestrian) and updating the position of the objects in consecutive frames in a video sequence. In our case, what is in our interest is not the prediction of the location of the object in the equirectangular image but the parameters of the vPTZ camera (pan, tilt and zoom) that hold it in its fov.

3.1.4 Evaluating tracking algorithms in 360° videos

As anticipated in 3.1.3 another limitation in the PTZ cameras is the lack of standard evaluation methodologies for PTZ tracking algorithms.

We have considered to evaluate our technique with the metrics described in [3]. Before to present these measures, c^t_{GT} and c^t_{PT} are defined as the center locations of the ground-truth target and the predicted target by the tracker at time t, respectively, A^t_{GT} and A^t_{PT} are the bounding boxes of the ground-truth target and the predicted target at time t. Lastly, c^t_{fov} is the location of the center of the image fov at time t.

The metrics are divided into those that measure the quality of target localization and those that measure the quality of the camera control. The metrics that deal with measuring the quality of target localization are The Center Location Error (CLE) and The Overlap Ratio (OR). The CLE is defined as:

$$CLE^t = |C^t_{GT} - c^t_{PT}| \tag{3.3}$$

and it should be as lower as possible. The OR is defined as:

$$OR^t = \frac{A^t_{GT} \cap A^t_{PT}}{A^t_{GT} \cup A^t_{PT}} \tag{3.4}$$

and it should be as near to 1 as possible. It is also known as the Jaccard index, and it indicates the intersection over the union score (IoU). The metrics that deal with measuring the quality of the camera control are the Target To Center Error (TCE) and the Track Fragmentation (TF). The TCE is defined as:

$$TCE^t = |C^t_{FOV} - c^t_{GT}| \tag{3.5}$$

and like the CLE it should be as lower as possible. The TF indicates whether the target is inside or outside the camera fov. The authors in [3] defined TF as:

$$TF^t = \begin{cases} 1 & \text{if } CLE^t \; is \; invalid \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

The TF indicates the percentage of invalid detections in the tracks. In fact TF is the sum of all TF^t divided by the number of processed frames.

Since CLE and TCE are expressed in pixels and since The TF indicates that a detection is invalid when it is outside the camera fov, these metrics are not general enough. In fact, CLE and TCE measurements depends on the image resolution of the projected views and on the zoom factor. TF is measured on the base of the vPTZ camera plane, whose dimensions (W x H) can be arbitrarily set.

For all the above reasons, we decided to not adopt the metrics suggested in [3], and to measure the mean absolute error (MAE) in degrees between the estimated pan, tilt, and zoom angles and the corresponding angles derived from the annotations. Furthermore, we preferred to define a more restrictive metric than the TF in [3] and we consider a detection as valid if it falls within the fov angle derived from the annotated target bounding-box. By indicating with a * the ground-truth values at time t and with α, β and γ the estimated pan, tilt and zoom angles in degrees, the new definition of TF is:

$$TF^t = \begin{cases} 1 & \text{if } |\alpha_t - \alpha_t^*| \le \frac{\gamma_t^*}{4} \ \& \ |\beta_t - \beta_t^*| \ \le \frac{\gamma_t^*}{2} \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

To make clearer the meaning of the TF, in Fig.3.4 is shown the possible range variation of the estimated angles α, β, γ with respect to the ground truth values. On equals term of angles, the red bounding box and the green one are the limit cases whose centers are still included within the pan range. If the actual zoom angle is equal to γ^*, the estimated detection defined by α, β overlaps with the ground-truth one if $|\alpha_t - \alpha_t^*| \le \frac{\gamma_t^*}{4}$ and $|\beta_t - \beta_t^*| \ \le \frac{\gamma_t^*}{2}$. Such differences take also into account the aspect ratio of the pedestrian detection, meaning that the tilt range is twice the pan range. In practice, by setting these thresholds on the pan and tilt angles, the true and estimated detections are constrained to overlap for at least 1/4 of their area. The criteria is similar to the intersection over union used in traditional tracking approaches to establish if a detection is valid or no.

On this basis we have adopted the MAE and modified TF to compare our tracker results with the work of [3] and others.

3.2 PTZ Tracking by detection

Tracking-by-detection has recently become one of the most common paradigm for tracking, and it has been employed in various domains [107, 108, 109, 110]. Tracking-by-detection relies on the availability of either a pre-trained or an online

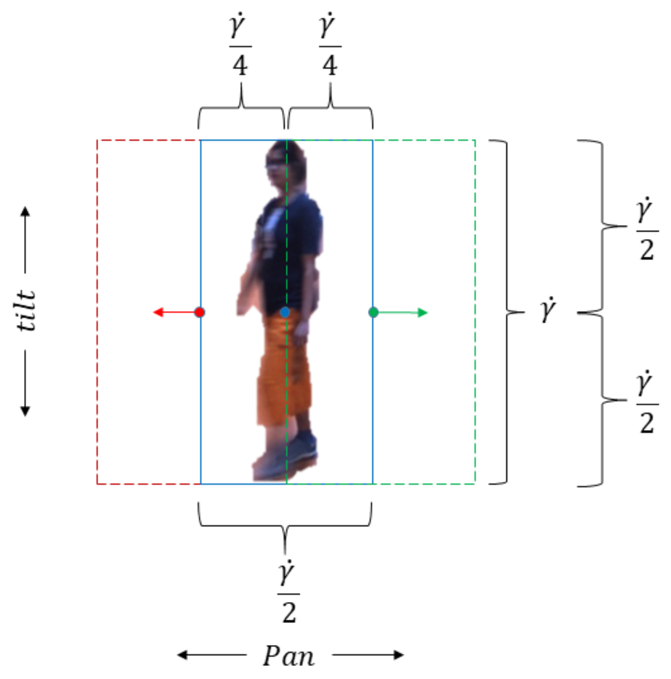


Figure 3.4: Relation between the zoom angle and the pan and tilt angles.

trained detector for the selected target. As for pedestrian tracking, there are numerous detectors in literature [55, 231, 117]. Despite in this thesis we employ pedestrian detectors, our frameworks are general and can be extended to track generic objects by choosing appropriate detection techniques, such as [114, 115, 5].

Many works follow the conventional approach that includes the extraction of hand-crafted features [6, 7] but they are unable to adapt to the complex, highly non-linear and time-varying variations of the target appearance. Newer approaches like ConvNets are able to learn the features from image data and to handle this complexity. In this section will be presented two works based on the tracking-by-detection paradigm:

- The first, *Pedestrian Tracking in 360 Video by Virtual PTZ Cameras* [15], exploits the use of a dynamic memory of the best past detections as appearance model. The target is detected by means of a set of pre-trained pedestrian detectors.

- The second, *Tracking-by-Detection in 360 degrees videos using pre-trained* *deep models*, proposes a pedestrian tracking-by-detection algorithm for 360° videos that uses a pre-trained deep ConvNet (Mask R-CNN) on images acquired by vPTZs. Mask R-CNN provides binary object segmentation mask that can be used to strengthen the target appearance modelby suppressing unnecessary background information.

Both these two algorithms aim at estimating the pan, tilt and zoom parameters required to control a virtual PTZ camera in such a way that the target is always at the center of the virtual camera view.

3.2.1 Pedestrian Tracking in 360 Video by Virtual PTZ Cameras

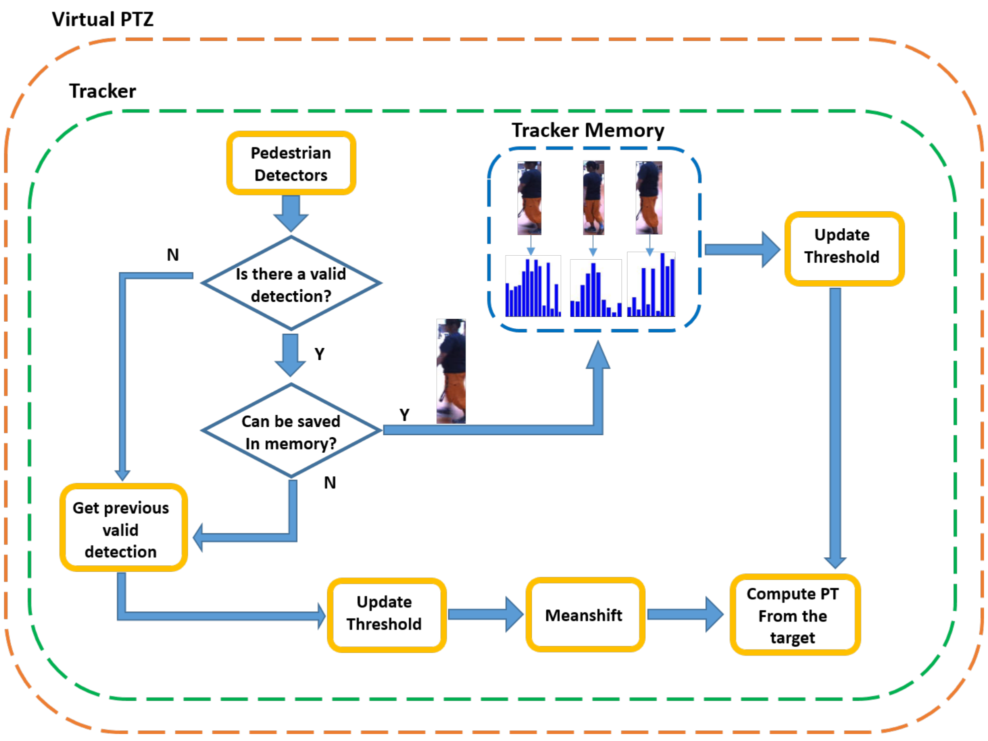


Figure 3.5: Pedestrian Tracking in 360 Video by Virtual PTZ Cameras

This framework processes equirectangular frames from 360° videos and, based on the current pan, tilt, zoom parameters, it projects the corresponding section of the world scene.

Figure 3.5 summarizes the main steps of our pedestrian tracker. As we have already mentioned, this tracker is based on pedestrian detectors. The main problem with pedestrian detectors is that they can detect numerous false positives in complex scenarios and may present several missing detections. To alleviate the problem of missing detections, it was used multiple detectors and, in particular, it was used the C4 pedestrian detector [117], the OpenCV HOG detector [55], and the OpenCV Haar Cascade pedestrian detector [114].

Among all the detections, it is selected the detection nearest to the center of the image (where the target is expected to be) and with the highest appearance similarity to the target. Whenever none of the available detections is similar to the target, or when no detection at all is available, then past detections are used to compute a confidence map, and the meanshift algorithm is adopted to locate the target in the confidence map.

Pan and tilt parameters are modified frame by frame to keep the target at the center of the virtual camera view. These parameters are obtained through the inverse mapping of the bounding box center as already described in Sec. 3.1.1.

In the following, are described the main components used in the algorithm.

The Appearance Model

The appearance model used in this tracker relies on histograms computed in the HSV color space. It only uses the Hue and Saturation channels. Inspired by the work in [232, 107], it was implemented a simple fixed-length dynamic memory that stores a predefined number of target color histograms. Furthermore, inspired by the work of [233], in order to obtain more discriminative color histograms, each pedestrian detection is divided into upper and lower parts. Given a target detection, it is extracted its color histograms and these are compared with the color histograms (for the upper and lower parts of the bounding boxes) stored in the dynamic memory by means of the Bhattacharyya distance [234]. This distance measures how different are two continuous or discrete probability distributions, and it is defined by the following equation:

$$d(H_1, H_2) = \sqrt[2]{1 - \frac{1}{\sqrt[2]{\overline{H_1 H_2} N^2}} \sum_I \sqrt[2]{H_1(I) H_2(I)}} \tag{3.8}$$

where H_1 and H_2 are the two color histograms to compare, N is the number of bins in each histogram, $\overline{H}$ indicates the mean value of the bin counts. This equation returns a value between 0 and 1, where 0 indicates that the two histograms are exactly the same.

To select the pedestrian detection that is more similar to the target, the color histograms of each detection are compared to the color histograms of the templates stored in the dynamic memory by Eq. 3.8. Hence, the detection with the lowest average Bhattacharyya distance is selected.

Dynamic memory management

The use of a dynamic memory of the past target detections allows to exclude false positive detections and to account for the varying appearance of the target. To decide how and when updating the dynamic memory, a threshold on the Bhattacharyya distance is learned online. In practice, the Bhattacharyya distances among the target detections are assumed to be distributed based on a Gaussian model. It is well known that the parameters of a Gaussian distribution μ and σ can be learned incrementally by means of the following equations:

$$\begin{aligned} \mu_{t+1} &= \frac{n\mu_t + d_{new}}{n+1} \\ \sigma^2_{t+1} &= \frac{n(\sigma_t^2 + \mu_t^2) + d^2_{new}}{n+1} - \mu^2_{t+1} \end{aligned} \tag{3.9}$$

where d_{new} represents the Bhattacharyya distance to add to the sample, and n is the sample size at time t.

To set a threshold T on the Bhattacharyya distances for deciding if a detection should enter in the dynamic memory, it is used the following equation:

$$T = \mu_t + 1.5\sigma_t. \tag{3.10}$$

The memory has a predefined size. When the capacity of the memory has already been reached, the memory is updated by replacing one of the template with the target detection that has distance lower than T. The selection of the template to remove from the memory is taken by selecting that having the highest distance to the current target detection.

Tracking-by-detection approach

Let's define $\mathbf{d_a}$ as the vector of the average Bhattacharyya distances of the pedestrian detections to the templates stored into the dynamic memory. This vector only represents appearance dissimilarities between the target and the pedestrians detected in the image. Since the tracker tends to keep the target at the center of the camera field of view, also the distances between the bounding box centers and the image center provides information about the target position in the image. Let's define $\mathbf{d_p}$ as the vector of the normalized Euclidean distances of the pedestrian detections to the image center. Normalization of the Euclidean distances is achieved by dividing the distances by the maximal distance to the center of the image. In this way, normalized Euclidean distances range in $[0, 1]$.

In order to consider both kind of distances, $\mathbf{d_a}$ and $\mathbf{d_p}$, it is considered a convex combination $\mathbf{d}$:

$$\mathbf{d} = \alpha * \mathbf{d_p} + (1 - \alpha) * \mathbf{d_a} \tag{3.11}$$

where α is a parameter in $[0, 1]$ that determines the weight of each kind of distances.

The target detection is selected among the pedestrian detections by considering the one with the lowest value of $\mathbf{d}$. To limit drifting of the tracker, such detection is considered valid only if $\mathbf{d}$ is lower than 0.5. The selected target detection will then be used to update the dynamic memory.

In some cases, no valid detection is available: none of the available detections has a distance $\mathbf{d}$ lower than 0.5, or no detections is returned by the detector. In these cases, the meanshift algorithm is used to perform tracking. The algorithm is initialized on the location of the most recent target detection. Meanshift is applied on a confidence map computed by means of the templates stored in the dynamic memory. In experiments, we noted that the adopted appearance model is very sensitive to illumination changes of the scene and, since the detections returned by the detectors have a large size, color histograms are affected by the presence of background within the bounding box. To alleviate this problem, we modify the histograms of each detection in memory by setting to 0 those bins that are largely present on the background. Such bins are found by analyzing the histogram of the regions surrounding the pedestrian detections.

Summary of the tracker algorithm

The pseudo code of this approach is shown in Algorithm 1. First, the tracker is initialized by setting the maximal number of templates (*number_memories*) that can be saved in memory. The appearance threshold is initialized with values μ and σ learned from a small portion of the training dataset. The *getBestDetection(frame)* function returns the detection with the smallest score obtained by Eq. 3.11 on the detections of the pedestrian detectors. The *addInMemory(target)* function checks if the detection can be used to update the memory. The *getLastTarget()* function returns the last location of the target. The *updateAppearanceThreshold()* function updates the appearance threshold T. *getconfidenceMap()* returns the average of the confidence maps of the color histograms saved in memory. $applyMeanShift(cm, target)$ executes the meanshift algorithm to find the target based on the color histogram. *getPT(target)* converts x-y coordinates of the bounding box of the target in pan and tilt values and *setPT(PT)* updates the pan and tilt of the vPTZ camera.

Algorithm 1

```
1: while $frame$ in $video$ do
2:     if $tracker$ not $initialized$ then
3:         $initializeTracker(number\_memories)$
4:         $initializeAppearanceThreshold(mean, variance)$
5:     else
6:         $detected \leftarrow false$
7:         $inMemory \leftarrow false$
8:         $bestDetection \leftarrow getBestDetection(frame)$
9:         if $bestDetection$ not $empty$ then
10:            $detected \leftarrow true$
11:            $target \leftarrow bestDetection$
12:            $inMemory \leftarrow addInMemory(target)$
13:            if not $inMemory$ then
14:                $target = getLastTarget()$
15:                $updateAppearanceThreshold(10)$
16:                $detected \leftarrow false$
17:            end if
18:        else
19:            $detected \leftarrow false$
20:        end if
21:        if not $detected$ and not $inMemory$ then
22:            $target = getLastTarget()$
23:            $cm = getconfidenceMap()$
24:            $applyMeanShift(cm, target)$
25:        end if
26:        $PT \leftarrow getPT(target)$
27:        $setPT(PT)$
28:    end if
29: end while
```

The framework presented has been implemented in C++ with the OpenCV 3.4.1 library. Experiments have been run on a machine equipped with 2 Intel i7-4770 Quad-Core CPUs (3.40GHz) and 16 GB of RAM. We performed experiments to assess the quality of our pedestrian tracking algorithm on the public dataset in [3]. Experimental results are presented in the Ch. 5.

3.2.2 Tracking-by-Detection in 360 degrees videos using pre-trained deep models

In this section a pedestrian tracking-by-detection algorithm for 360° videos that uses a pre-trained deep ConvNet (Mask R-CNN [127]) on images acquired by vPTZs is proposed. Mask R-CNN provides binary object segmentation mask that can be used to strengthen the target appearance model by suppressing unnecessary background information. Mask R-CNN is able to generate the segmentation masks for each instance of an object in an image as shown in fig. 3.6. Kalman filter is adopted to improve the tracker accuracy whenever a partial occlusion arises or the target detection fails.

In recent years, deep Convolutional Neural Networks (ConvNets) have significantly improved vision-based systems [154, 235]. Models such as Mask R-CNN [127] have shown impressive capabilities in generating accurate segmentation masks of objects depicted in an image. As an example, Mask R-CNN, trained on COCO dataset with ResNet-101 backbone, can detect 80 different classes of objects. Considering the accuracy reached by these models, the main contribution of this method is a tracking-by-detection algorithm for 360° videos that combines the use of vPTZ cameras and Mask R-CNN to track pedestrians.

We show that, by using a stronger pedestrian detector based on deep learning techniques, it is possible to achieve better results on the publicly available dataset [3]. Our experiments also show that the use of Kalman filter is viable for the vPTZ pan and tilt parameters but not for the zoom factor. We experimentally found that the continuous correction of the zoom factor may prevent the Mask R-CNN net to properly detect the pedestrians, and negatively affect the tracking results.

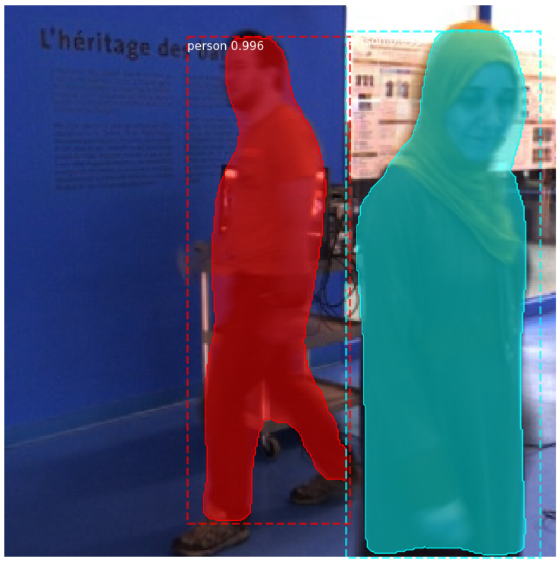


Figure 3.6: Segmentation masks of two pedestrians obtained by the Mask R-CNN

The Proposed Pedestrian Tracker

Fig. 3.7 shows an image of the proposed framework.

The tracker is initialized at time $t = 0$ by specifying *pan*, *tilt*, *zoom* values for the vPTZ to permit to acquire an image in which the target is centered.

The target's appearance model is computed as explained in sec. 3.2.2 and stored to be used as future reference.

Mask R-CNN is used to detect pedestrians (Detection Module) in the current vPTZ camera view. As detailed in Sec. 3.2.2, an appearance descriptor extracted from these detections (Appearance Module) is compared to the target appearance model. The comparison allows to identify the most similar detection to the target. If no pedestrian is detected or none of the detection resembles the target, a failure in the detection process occurs and the Missing and Occlusion Handling Module is invoked. Kalman filter, see Sec. 3.2.2, is used to correct and predict the new pan, tilt, zoom vPTZ parameters.

If instead, the target has been detected and recognized in the vPTZ camera view, the Update Module updates the target location in terms of the parameters *pan*, *tilt* and *zoom* of the vPTZ camera.

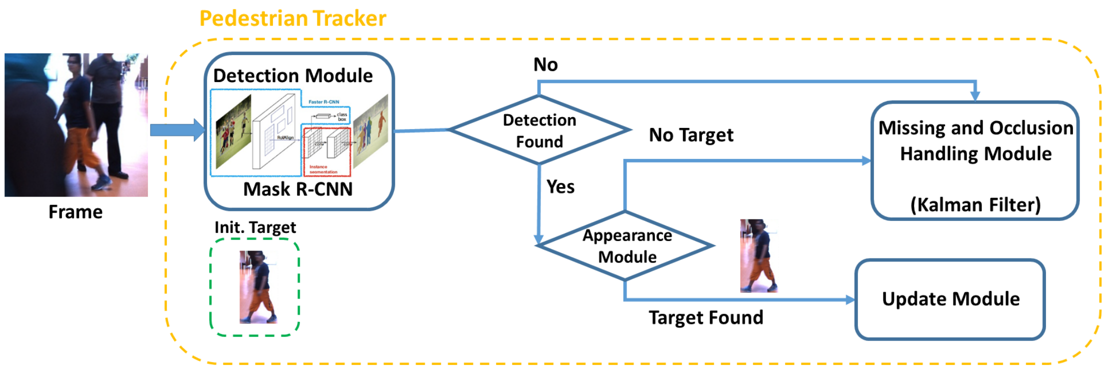


Figure 3.7: Pedestrian Tracking in 360 Video by Virtual PTZ Camera

Detection and Appearance Modules

In the detection module, it is used Mask R-CNN [236] to detect pedestrians at time t by using the pan, tilt, zoom parameters predicted at time $t-1$. In the appearance model, it is adopted the same approach of [3, 15], which is based on the histograms computed in the HSV color space of our detections.

Since the tracker needs to keep the target at the center of the vPTZ FOV, the distance between the detection bounding box center and the vPTZ image plane center helps to discriminate among several candidate detections.

We define $\mathbf{d_p}$ as the vector of the normalized Euclidean distances of the pedestrian detections to the image center. Normalization of the Euclidean distances is achieved by dividing them by the distance of the top-left corner to the image center. Normalized Euclidean distances range in $[0, 1]$, with 0 indicating that the bounding box is centered in the image.

We further define $\mathbf{d_a}$ as the vector of the distances of the color histograms of pedestrian detections to the color histogram of the first target bounding-box saved at time $t = 0$. Such distance is computed by means of the Bhattacharyya distance [234]. The Bhattacharyya distance measures how different are two probability distributions (in our case, color histograms in the HSV color space) and ranges in $[0, 1]$ with 0 indicating that the two histograms are the same.

$\mathbf{d_p}$ and $\mathbf{d_a}$ have size equals to the number of detections returned by the R-CNN Mask at time t. Assuming that n is the number of pedestrian detections found at time t, these two vectors have a dimension equals to $n \times 1$. The convex combination of these two vectors in Eq. 3.12 is defined as:

$$\mathbf{d} = \alpha * \mathbf{d_p} + (1-\alpha) * \mathbf{d_a} \tag{3.12}$$

where α is a parameter in $[0,1]$ and determines the weight of each kind of distance. Since the network can detect more than one pedestrian in the image acquired by the vPTZ, it is necessary to associate a detection to the target. To select the pedestrian detection that is more similar to the target we select the detection with the lowest value of $\mathbf{d}$ but that is greater than a specific threshold μ.

Missing and Occlusion Handling Module

A missing or occlusion condition occurs in only two cases: if the Detection Module has not found pedestrians or if the target has not been recognized among the various pedestrian detections by the Appearance Module. The system uses a progressive zoom out strategy to increase the vPTZ camera FOV of a certain value λ to allow the Mask R-CNN to have a greater chance of detecting the target. In addition to the progressive zoom out strategy, is used the Kalman filter [73] for the prediction of the pan, tilt, zoom parameters to use at the next time. We define the state vector of our vPTZ camera at time t as $\mathbf{x_t}$. $\mathbf{x}$ is a 6×1 vector containing the vPTZ camera parameters (pan, $tilt$, $zoom$) with their first-order derivatives ($\dot{pan}$, $\dot{tilt}$,$\dot{zoom}$).

We consider a discrete time dynamical system governed by Eq. 3.13.

$$\mathbf{x_t} = A\mathbf{x_{t-1}} + w_{t-1}, \quad \mathbf{z_t} = H\mathbf{x_t} + v_t \tag{3.13}$$

The Process Model relates the state at a previous time $t-1$ with the current state at time t. The matrix A is called state transition matrix and w represents the normally distributed process noise that affects the system at time $t-1$. We assume that w has a multivariate normal distribution with mean 0 and covariance matrix Q, $w_{t-1} \sim \mathcal{N}(0, Q)$. We obtain a vector of measurements z_t at time t , where $\mathbf{z_t}$ is given by the Eq. 3.13. It relates the current state to the measurement

$\mathbf{z}$ with the matrix H. H is actually a transformation matrix that transforms state space to measurement space. v is the normally distributed measurement noise with mean 0 and covariance R, $v_t \sim \mathcal{N}(0, R)$. The matrices A, H, Q, and R are all assumed to be known, they are initialized at time $t = 0$, although the Kalman filter can be extended to simultaneously estimate these matrices along with $\mathbf{x_t}$.

The process noise covariance matrix Q, and measurement noise covariance matrix R, can be constructed to get the best performance [73]. However, in real applications we do not know the real statistics of the noises and the noises are often not Gaussian. Common practice is to set Q and R on the basis of the uncertainty of the initial guess: the more uncertain the initial guess for the state is, the larger the initial error covariance should be. A basic way to think of Q and R is that they are weighting factors between the prediction states and the measurement in Eq. 3.13. So, we set Q and R as diagonal matrices ($n \times n$, where n is the number of the state variables in $\mathbf{x_t}$) with constant values of 10^{-3} and 10^{-5} respectively.

Even with this tracker we performed experiments on the public dataset in [3]. Experimental results are presented in the Ch. 5.

3.3 Bayesian Approach

Particle filter methods provide a convenient approach to approximate the posterior distributions of object state when a system is non-linear and when the noises are not-Gaussian. For this reason Particle filter has been successfully applied into a wide range of fields such as computer vision, robotics, signal processing, economics, etc. In 1998, Isard and Blake [68] proposed a novel algorithm for visual tracking applications called *Condensation* (another name of particle filter) by proving its superiority in awkward circumstances such as object occlusions, background clutters and multi-object tracking, that are very common issues in real world applications.

The Particle filter approach tries to consider the visual tracking problem as an inference problem under Bayesian framework. In this framework, state space models about object state transition and observation models are constructed, and the posterior probability density of object states is recursively estimated by the use of the prior probability and observations available.

The following method proposes to formulate the visual tracking problem as the one of selecting, at each time, the vPTZ camera to foveate on the target from the unlimited set of simultaneously generated vPTZ camera views. Assuming that the selected vPTZ camera is a stochastic variable, this method proposes to model the posterior distribution of the underlying stochastic process by means of a set of particles each representing a vPTZ camera view.

3.3.1 Particle Filtering for Tracking in 360° Videos using virtual PTZ Cameras

The main idea of this method was to generate a number of N vPTZ cameras from a single 360° video and to select the vPTZ that can best track the target at a given time based on particle filter algorithm.

In this method rather than formulating the tracking problem as the one of controlling a single vPTZ camera, as in [15], we assumed that the vPTZ camera to foveate on the target at a time instant is a random variable, and we approximate the posterior distribution of the underlying stochastic process by means of a discrete set of vPTZ camera views. To model the posterior distribution over vPTZ camera views we adopt the particle filter framework in which each particle is a vPTZ camera with specific pan, tilt and zoom parameters (see Fig. 3.8).

For each frame of the 360° video, the tracking algorithm involves three main phases: prediction, updating and re-sampling. In the prediction step, the algorithm samples vPTZ camera views; in the update phase, it measures the likelihood that each generated vPTZ camera view can track the target, and updates the posterior distribution accordingly. The re-sampling procedure is applied to avoid particle degeneration. The vPTZ camera view to track the target is found as expected value of the posterior distribution.

The idea of using particle filter with vPTZ cameras is, at the best of our knowledge, new and not yet covered in other works. For the sake of clarity, we note here that our proposed application of the particle filter to model the posterior distribution of vPTZ camera views differs from the application of the same technique to model the posterior distribution of the target's location on the equirectangular image. Indeed, in the framework, a particle does not directly represent a patch of the equirectangular image. Instead, it represents the projection of the spherical surface onto a tangent plane accounting, in this way, for the sphericity of the 360° image. Furthermore, we focus on the problem of tracking a single target and, hence, we do not apply any data association technique as generally required in multi-target tracking framework.

To validate the pedestrian tracker, it was used the publicly available dataset of spherical videos in [3]. The experiments show that our technique is viable and achieves state-of-the-art performance.

3.3.2 Sequential State Estimation Problem

Given a series of observations from time 1 to t, the sequential state estimation problem concerns the estimation of the current optimal state based on the observations up to time t, which in other words means estimating the posterior probability density of the state itself.

Given the state x_t and x_{t-1} at time t and $t-1$ respectively, and the stochastic

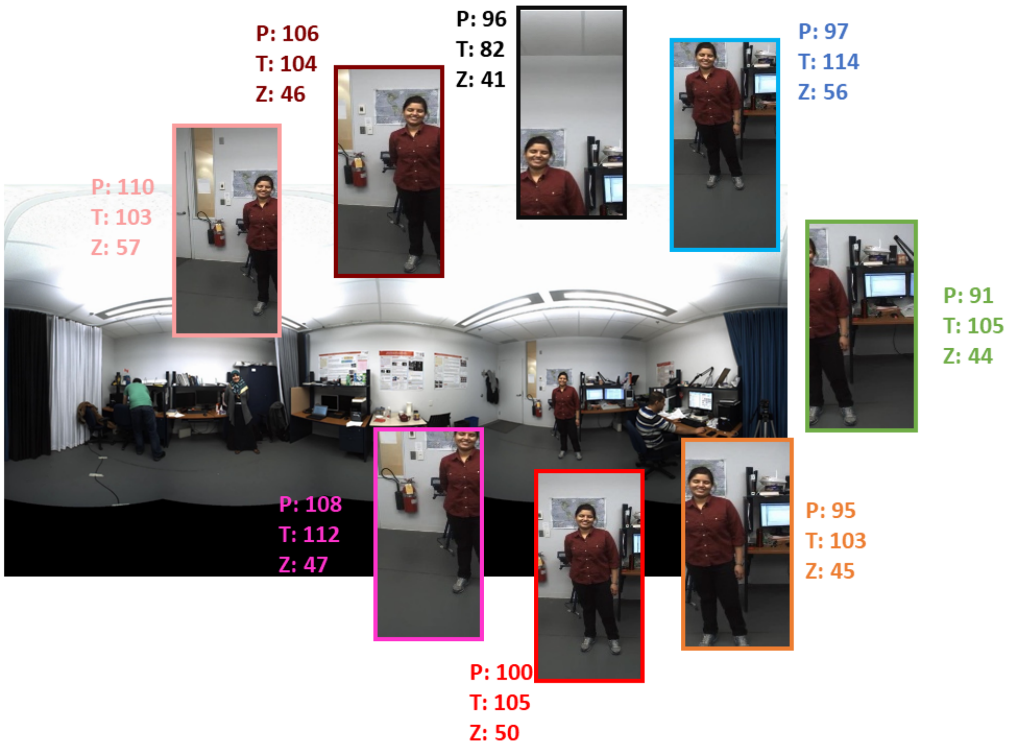


Figure 3.8: The image shows an equirectangular image and a set of views acquired by different vPTZ cameras with the reported parameters.

noise d_{t-1}, the transition state function is defined f_t as:

$$x_t = f_t(x_{t-1}, d_{t-1}). \tag{3.14}$$

The presence of the random variable d_{t-1} induces a conditional probability density function over the state x_t.

Let's define y_t as the observation at time t and v_t as the stochastic noise with which we observe the state. The dependency between y_t, x_t, v_t by means of the function g_t is defined as:

$$y_t = g_t(x_t, v_t). \tag{3.15}$$

Since v_t is a random variable, it induces a probability density function, known as likelihood, that is indicated as $p(y_t|x_t)$.

The states are hidden in the sense that they are not observable and all we can know about the system is through its observations at each time instant.

To solve the sequential state estimation problem we decided to adopt the algorithm of the particle filter that implements a Bayesian filter and estimates the states of the system on the basis of an indirect observation of the same.

3.3.3 Particle filter

In its most generic sense, tracking is the problem of estimating a hidden state sequence $X_t = \{x_i\}_{i=1}^t$, from a sequence of noisy and possibly nonlinear observations $Y_t = \{y_i\}_{i=1}^t$ from time 1 to t. We assume the state sequence X_t follows a first order Markov chain, that is, at each time step, the state x_t only depends on the state at the previous time step x_{t-1} rather than on the entire state history X_t; we further assume that the observations Y_{t-1} are conditionally independent up the state at time t.

Particle filter implements a recursive Bayesian state estimator to solve the above mentioned problem by using a discrete sample set of weighted particles to recursively approximate the posterior distribution of the estimated state. The state x_t of an object is defined with a set of variables k_i, that is $x_t = \{k_1, ..., k_n\}$, called *state vector*, to describe a dynamic system at a given instant t. As t varies, the *state vector* describes a trajectory in the state-space that is called the trajectory of the system.

In visual tracking applications, the state can represent the position of the target at a specific time t. The observation is often an appearance descriptor of the target.

The particle filter main steps are: prediction and update.

- Prediction: the algorithm uses the previous state to predict the current state;
- Update: the algorithm uses the current observation to correct the state estimate.

Assuming that the initial state distribution $P(x_0)$ is known, under the first order Markov chain assumption, the goal of particle filter is to estimate the posterior state distribution $p(x_t|y_{1:t-1})$.

Given all observations up to time $t - 1$, $\{y_1, ..., y_{t-1}\}$, the prediction phase uses the $p(x_t|x_{t-1})$ to predict the posterior state distribution $p(x_t|y_{1:t-1})$ by the recursive equation:

$$p(x_t|y_{1:t-1}) = \int_{x_{t-1}} p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1} \tag{3.16}$$

where $p(x_{t-1}|y_{1:t-1})$ represents the posterior distribution of x_{t-1} given all observations up to $t-1$, and $p(x_t|x_{t-1})$ is the state transition probability.

At time t, the observation y_t becomes available and the state x_t can be updated by using the Bayesian Filter formula:

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})} \tag{3.17}$$

where $p(y_t|x_t)$ is known as *observation likelihood.* In the above equation, it is in general difficult to evaluate the *normalization factor* $p(y_t|y_{1:t-1})$ in closed form. For this reason, we consider $p(x_t|y_{1:t}) \propto \ p(y_t|x_t)p(x_t|y_{1:t-1})$.

To overcome the complexity of such estimations, in particle filtering, the posterior density is approximated by a discrete set of particles $\{\tilde{x_t}^i\}_{i=1}^N$ each with a weight w_t^i. Since it is difficult to sample from the posterior distribution, particles $\tilde{x_t}^i$ are drawn from a proposal distribution $Q(x_t|x_{1:t-1}, y_{1:t})$.

At each time, when a new observation becomes available, the posterior distribution is updated by modifying the particle weights as follows:

$$w_t^i = w_{t-1}^i \cdot \frac{p(y_t|\tilde{x_t}^i)p(\tilde{x_t}^i|\tilde{x}_{t-1}^i)}{Q(\tilde{x}_t|x_{1:t-1}, y_{1:t})}. \tag{3.18}$$

Weights are then normalized to sum 1. To avoid particle degeneration, particles are resampled based on their importance weights.

3.3.4 vPTZ Camera Filtering for Tracking

In this section it is presented our tracker for 360° video. The framework is based on the particle filter algorithm and the possibility to generate a number of N vPTZ cameras from a single 360° video.

In this approach, we define x_t as the vPTZ camera that can best track the target at time t; we also define y_t as the corresponding observation at time t, for example the target appearance descriptor extracted from the view of the vPTZ camera x_t.

We model x_t as a random variable and adopt particle filter to formulate the underlying stochastic process. We approximate the posterior distribution over the vPTZ cameras x_t by means of a set of particles $\{\tilde{x_t}^i\}_{i=1}^N$. Each particle $\tilde{x_t}^i$ is drawn from a proposal Gaussian distribution and represents a different vPTZ camera characterized by its own pan, tilt and zoom parameters. Some of these cameras will be directed towards non-interesting areas for the tracking purpose while others will be directed towards the target.

Each vPTZ camera $\tilde{x_t}^i$ is a particle weighted by w_t^i. The weights of the particles are updated as described in Eq. 3.18 and normalized to sum 1.

The vPTZ camera x_{t+1} to track the target is computed as:

$$x_{t+1} = \sum_{i=1}^{N} w_t^i \cdot \tilde{x}_t^i. \tag{3.19}$$

The above described steps are iterated frame-by-frame to keep the target at the center of the vPTZ camera FOV. To prevent the vPTZ cameras degenerate, multinomial resampling is used.

3.3.5 State Model

The state of a vPTZ camera is described with five parameters: α, β, γ, which represent the pan, tilt, zoom angles, and the velocities of pan and tilt, that are $\dot{\alpha}$ and $\dot{\beta}$. We assume the zoom angle varies with zero velocity because its variations are in general smoothed. In particular, the state x_t is described as:

$$x_t = \left[\alpha,\ \beta,\ \gamma,\ \dot{\alpha},\ \dot{\beta}\right]$$

We also assume that our system propagates particles according to a first order motion model specified by Eq. 3.20 where δt is a constant value:

$$x_t = Ax_{t-1} + d_{t-1}, \text{ with } A = \begin{bmatrix} 1 & 0 & 0 & \delta t & 0 \\ 0 & 1 & 0 & 0 & \delta t \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.20}$$

3.3.6 Observation Model

The first target detection, extracted at specified *pan*, *tilt*, *zoom*, is used to initialize the appearance model y_1 of the tracker

Inspired by the work of [233], to obtain more discriminative histograms, the image is divided into upper and lower parts and two different histograms are extracted from them. Our appearance model relies on these histograms computed on the Hue and Saturation channels of the image.

The observation likelihood $p(y_t|x_t)$ is modeled by a Gaussian distribution (centered in 0 and of variance σ_a^2) over the Bhattacharyya distance [234] of the appearance models y_t and y_1.

The Bhattacharyya distance is defined in 3.8.

When an occlusion arises (for example, the target is occluded by another pedestrian), the likelihood of the particles decreases and, consequently, particles begin to die (i.e., particles have low or uniform weights). To account for such particle impoverishment, a resampling procedure is used to refresh the set of particles. We stress here that, since our tracking approach focuses only on a single target and no pedestrian detector is used during tracking, our algorithm does not require any data association procedure.

Experimental results are presented in the Ch. 5 and were performed on the public dataset in [3].

Chapter 4

Methods for Human Behavior Characterization

This Chapter treats methods to analyze humans' movements, such as human activity recognition, and to analyze humans' behaviors represented through multi-dimensional time-series. Characterization of human behaviors is a challenging problem in Computer Vision and it represents a key application in several fields, such as security, natural interfaces, gaming and assisted living, to name a few.

The goal of human activity recognition is to automatically detect and analyze human activities from the information acquired from sensors (e.g from video cameras, Kinect, etc.).

In the literature there is not a unique definition of the notion of *activity*. However, common practice is a to assume activities as aggregations of actions, which again are understood as aggregations of atomic operations [237, 238, 239]. In other words, activities are hierarchically structured into actions, as for example the activity of *preparing coffee* can be structured in the actions *fill the water container*, which can be further decomposed into atomic operations as *open the water tap* [237].

The application of HAR involves a better understanding of the patterns of human movements. As suggested in [240] the target tracking problem and the recognition of atomic activities indicate a strong dependence and should be considered into a single coherent framework to allow the construction of more accurate surveillance systems. For instance, understanding that a person is performing an action like walking rather than another would permit to a tracker to use different strategies to improve its track accuracy.

4.1 Hop: Histogram of Patterns for Human Action Representation

In this framework we proposed to represent time-series of descriptors by means of distributions of frequent sequential patterns of different length for action classification. We define a sequential pattern as a series of data descriptors indexed in time order, and a frequent pattern is one that occurs many times in the data [13].

A classical approach to represent actions is *Bag Of Visual Words (BoVW)* [20, 21, 22, 23, 24]. In BoVW, an action is represented as a distribution of image/video patches (visual words). The codebook of visual words is generally computed by clustering algorithms, i.e. k-means ([25, 26, 27, 28]). To consider the dynamics of visual information in a time-series within BoVW, spatio-temporal descriptors extracted from fixed-length cuboids [21, 22, 24] or multi-scale time windows [145] have been used. Visual feature dynamics are especially useful for discriminating actions that share similar body poses but show different temporal evolution; as an example, *sit down* and *get up* are actions sharing similar body poses, but these poses appear in different time order.

In contrast to the classical BoVW approach, we describe an action by means of frequent sequences of visual descriptors, thus focusing more on the body motion dynamics rather than actual body poses. Fig. 4.1 gives an overview of the proposed *Bag-of-Frequent-Sequential-Patterns* approach. In this approach, the codebook of frequent sequential patterns is computed by means of a modified apriori algorithm [149, 241]. Our implementation of the apriori algorithm allows us to calculate frequent patterns of different lengths, which represent different levels of body motion details. While in general clustering algorithms group elements based only on pairwise element similarities, our technique considers both similarity and frequency of the elements when learning a codebook of frequent sequential patterns. This allows us to ignore infrequent patterns that might be less informative or even confusing for classification purposes.

To summarize, our contribution in this work is twofold:

1. It represents actions by multinomial distributions of frequent sequential patterns;

2. It proposes an apriori algorithm-based learning approach for codebook of frequent sequential patterns.

We demonstrate our approach in the context of 3D skeleton-based action classification [11]. The proposed framework can be easily extended to other kinds of visual descriptors such as histograms of STIP features [21] or HOG [55].

Experimental results are performed on the Microsoft Research Cambridge-12 (MSRC-12) gesture dataset [242] in cross-subject validation. Our technique achieves state-of-the-art accuracy values as it will be presented in the Ch. 5.

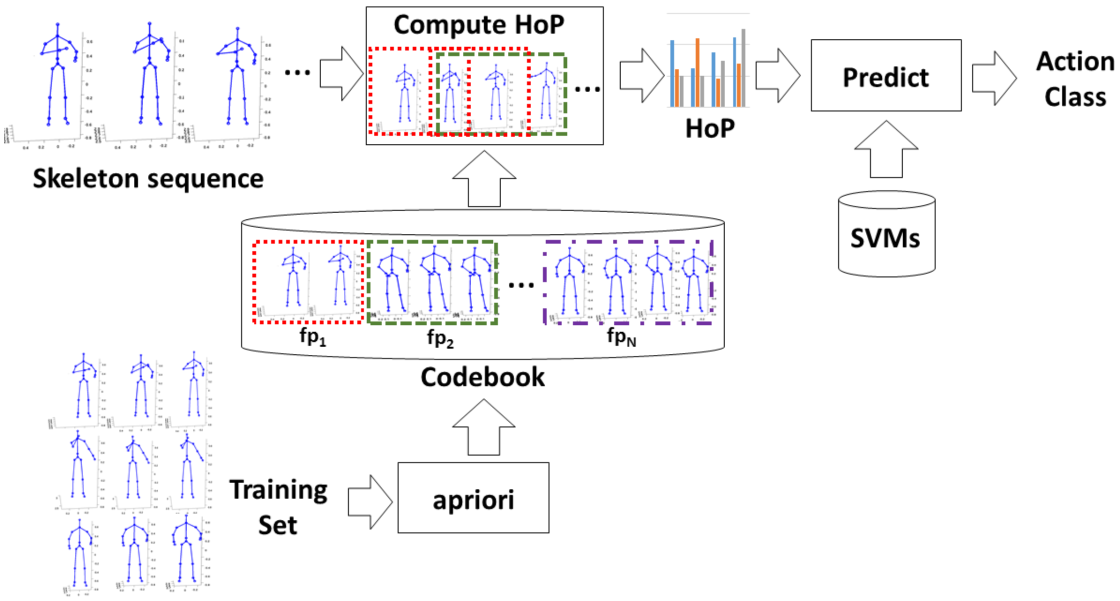


Figure 4.1: *Bag-of-Frequent-Sequential-Patterns:* a test sequence is encoded in terms of frequent sequential patterns (fp$_1$, fp$_2$, ..., fp$_N$) by means of vector quantization; hence, a histogram of frequent sequential patterns is computed and used to predict the action class based on 1-vs-1 SVMs. In the proposed approach, the codebook is learned by a modified apriori algorithm on the training set.

4.1.1 Representation by Histogram of Frequent Patterns

As shown in Fig. 4.1, a time-series is represented as a histogram of frequent patterns by matching subsequences with the patterns stored in the codebook.

Frequent patterns may be found by data mining techniques, such as the apriori algorithm proposed for transactional databases [243]. In such kind of applications, a pattern $C^{(k)}$ is a set of k items from an alphabet $\mathscr{A}$, and the problem is that of finding the longest frequent patterns in the database.

Since in transactional databases there is no need of considering the order of the items within the patterns, the method is not appropriate for sequential data, such as time-series, and requires some modifications in order to calculate frequent ordered item-sets. Modified apriori-algorithm for sequential data have been proposed in [149, 13, 243]. In particular, the method in [149] deals with the discovery of reduplication of ASL within a single data stream. As we will detail next, we borrow some of the ideas in [149] and adapt them to the learning (rather than discovering) of sequential patterns from a set of time-series.

4.1.2 Codebook of Frequent Patterns

The main idea behind apriori-like algorithms is that a pattern $C^{(k)}$ is frequent if and only if each pattern $C^{(k-1)} \subset C^{(k)}$ is frequent as well. Therefore a frequent pattern $C^{(k)}$ may be generated iteratively by extending a pattern $C^{(k-1)}$ with an item $i \in \mathscr{A}$, and ensuring that the generated pattern is composed of only frequent sub-patterns.

At the k-th iteration, apriori-like algorithms consist mainly of three steps:

- Generation of candidates of length k by frequent patterns of length $k-1$;
- Counting of candidate frequencies;
- Removal of infrequent patterns.

Infrequent patterns have a frequency count lower than a predefined threshold ψ.

We modified these steps to adapt them to the processing of sequential data. Alg. 2 shows the work-flow required to discover frequent patterns from training data $\mathscr{D}$. The algorithm generates frequent sequential patterns $\underline{C}^{(K_M)}$ of maximal

Algorithm 2 Learning a codebook of frequent sequential patterns

```
1: function CODEBOOK = CODEBOOKLEARNING($\mathscr{D}$, $K_M$)
2:     $k \leftarrow \tau$
3:     codebook $\leftarrow \emptyset$
4:     $\underline{C}^{(k)} \leftarrow$ generateCandidatePatterns($\mathscr{D}$, k)
5:     while $k < K_M$ do
6:         $k \leftarrow k + 1$
7:         $[\underline{C}^{(k)}, \underline{fp}^{(k-1)}] \leftarrow$ newCandidatePatternGeneration($\underline{C}^{(k-1)}$)
8:         codebook $\leftarrow$ codebook $\cup\ \underline{fp}^{(k-1)}$
9:         $\underline{C}^{(k)} \leftarrow$ duplicatesRemoval($\underline{C}^{(k)}$)
10:        getFrequencies($\underline{C}^{(k)}$, $\mathscr{D}$)
11:        $\underline{C}^{(k)} \leftarrow$ infrequentPatternsRemoval($\underline{C}^{(k)}$)
12:    end while
13:    codebook $\leftarrow$ codebook $\cup\ \underline{C}^{(K_M)}$
14: end function
```

length K_M. At the k-th iteration, $\underline{C}^{(k)}$ is a set of patterns $C_i^{(k)}$ with $i \in [1, \ldots, N_k]$, where N_k represents the number of frequent sequential patterns of length k that have been found in data $\mathscr{D}$. Each $C_i^{(k)}$ is an ordered sequence of feature descriptors $c_{i,j}$, i.e. $C_i^{(k)} = [c_{i,1}, c_{i,2}, \ldots, c_{i,k}]$. The set *codebook* stores frequent sequential patterns of different-length. The set $\underline{fp}^{(k-1)}$ stores frequent sequential patterns of length $k-1$ that cannot be used to generate longer patterns.

Candidate Pattern Generation: In the classical apriori algorithm [241], the initial set of items (alphabet $\mathscr{A}$) is known. In our application, this initial set is unknown and we start the algorithm with all possible windows of minimal length τ extracted from the data streams with a sliding window approach. We refine such initial set of candidate patterns $\underline{C}^{(\tau)}$ by pruning the duplicated and infrequent ones as detailed later.

Candidate Pattern Frequencies: Given a set of candidate patterns $\underline{C}^{(k)}$ and data $\mathscr{D}$, we need to count how many times each candidate pattern occurs in the data. In contrast to the classical apriori algorithm, this method entails the processing of non categorical data; therefore we need a strategy to establish approximate matches between candidate patterns and data. In particular, each candidate pattern $C_i^{(k)}$ has to be compared against temporal windows extracted from data and of the same length as the considered pattern. Let us assume for a moment that $\mathscr{D}$ contains only one sequence, i.e. $\mathscr{D} = [d_1, d_2, \dots d_N]$, and consider a pattern $C_i^{(k)} = [c_{i,1}, c_{i,2}, \dots, c_{i,k}]$. We consider a sliding window $W_t = [d_t, d_{t+1}, \dots, d_{t+k-1}]$. The similarity between the candidate pattern and the temporal window W_t is measured by the following similarity score:

$$s(C_i^{(k)}, W_t) = \frac{1}{k} \cdot \sum_{j=1}^{k} e^{-\lambda \cdot ||c_{i,j} - d_{t+j-1}||_2} \tag{4.1}$$

where λ is a scaling parameter that multiplies the per-item squared Euclidean distance. When this score is greater than a threshold ϵ, it is possible to establish a match between the pattern and the window, and increment the candidate pattern frequency. For each pattern, we keep track of the matched temporal windows by considering the list $\underline{W}^{C_i} = \{W_j\}_{j \in J}$.

New Candidate Pattern Generation and Codebook Learning: Let us consider two frequent patterns $C_1^{(k-1)} = [c_{1,1}, c_{1,2}, \dots, c_{1,k-1}]$ and $C_2^{(k-1)} = [c_{2,1}, c_{2,2}, \dots, c_{2,k-1}]$ such that $c_{1,j} = c_{2,j-1}$ $\forall j \in [2, k-1]$. Following [149], a candidate frequent pattern of k items can be defined as $C^{(k)} = [C_1^{(k-1)}, c_{2,k-1}]$. Fig. 4.2 sketches the new candidate pattern generation procedure.

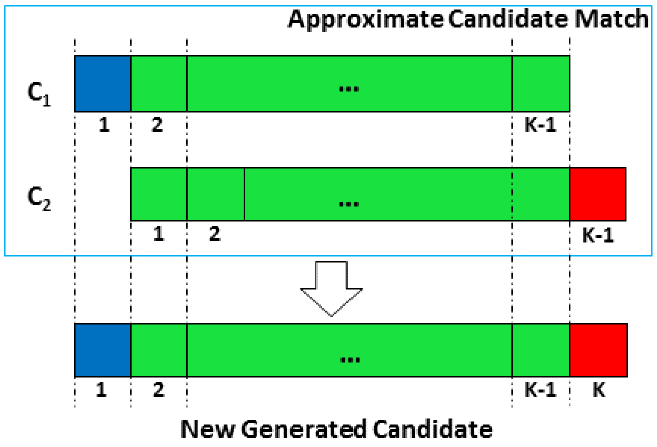


Figure 4.2: The figure illustrates the idea behind the candidate pattern generation process. The new generated candidate is formed by concatenating the first item of C_1, the items shared by both C_1 and C_2, and the last item of C_2.

This candidate generation procedure would work in case of exact match of the items. In our implementation, we establish approximate matches between candidate patterns $C_1^{(k-1)}$ and $C_2^{(k-1)}$ when all corresponding items score a similarity greater than ϵ. By defining the following binary variable:

$$m(C_1^{(k-1)}, C_2^{(k-1)}) = \prod_{j=2}^{k-1} (e^{-\lambda \cdot ||c_{1,j} - c_{2,j-1}||} \geq \epsilon), \tag{4.2}$$

if $m(C_1^{(k-1)}, C_2^{(k-1)})$ is equal to 1 then an approximate match between the two candidate patterns can be established.

In contrast to [149], where the items of each frequent pattern comes from the data stream, we learn a pattern model by means of the lists of matched windows of the two candidate patterns, respectively $\underline{W}^{C_1}$ and $\underline{W}^{C_2}$. The new generated pattern will have the form $C^{(k)} = [\mu_1, \mu_{2:k-1}, \mu_k]$ where μ_1 is the expected value of the first item of $C^{(k)}$ and is computed by averaging the first elements of the windows in $\underline{W}^{C_1}$; $\mu_{2:k-1}$ are expected values of subsequent items in the pattern $C^{(k)}$ and are calculated by considering both the items of windows in $\underline{W}^{C_1}$ and windows in $\underline{W}^{C_2}$; finally, μ_k is the expected value of the last item in $C^{(k)}$ and is computed by averaging the last elements of the windows in $\underline{W}^{C_2}$.

Whenever a candidate pattern of length $k-1$ does not contribute to generate candidate patterns of length k, and its frequency is greater than a threshold ψ, then the pattern is stored into the codebook.

Removal of Duplicated and Infrequent Candidate Patterns: After the generation step, a pairwise comparison of candidate patterns is carried on. Each pair of candidates with a similarity score greater than ϵ is replaced by a new candidate generated averaging the lists of matched windows. Such kind of pruning is necessary to deal with approximate matches between data and patterns. To focus on frequent patterns, candidate patterns with a frequency count smaller than a threshold ψ are considered infrequent and, hence, pruned.

4.1.3 Histogram of Frequent Patterns

Provided with a codebook of N frequent sequential patterns $\{C_i\}_{i\in[1,N]}$ of different length, we aim at representing a time-series $V = \{y_1, y_2, \ldots, y_v\}$ as a histogram of frequent patterns (HoP) by performing vector quantization (VQ) [24]. For each frame in V and for each pattern C_i in the codebook, we consider a subsequence of V that starts from the current frame, and of length equal to that of the considered pattern C_i. We compare each window to the patterns by the score in Eq. (4.1) and only increment the bin of the histogram that corresponds to the pattern achieving the highest similarity (i.e. we apply hard coding).

At the top of Fig. 4.3, a sample of the action class *Push-Right* is shown. The bar under the sequence indicates which patterns in the codebook have been detected in the sequence (each color corresponds to a different pattern); the patterns are represented under the bar while, at the bottom of the figure, the histogram of patterns is plotted.

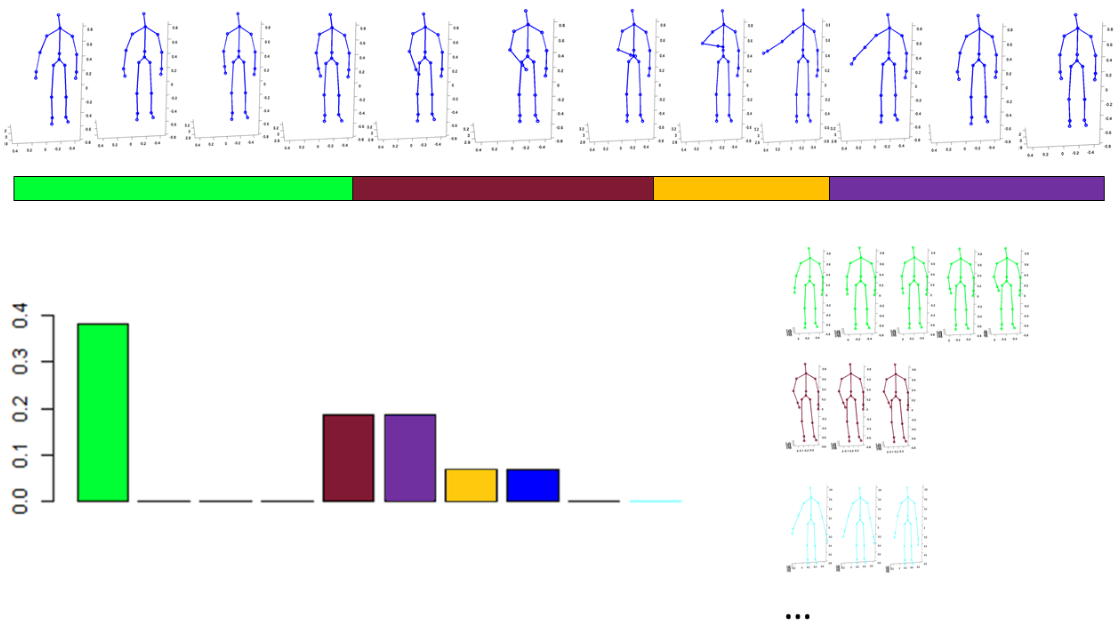


Figure 4.3: The figure illustrates the HoP of a sample of the $Push - Right$ class in terms of frequent patterns learned by our apriori algorithm. In the figure, only few distinctive skeletons of the sequence and of the patterns are shown.

This method is validated on the Microsoft Research Cambridge-12 (MSRC-12) gesture dataset [242]. In this dataset we achieves state-of-the-art accuracy values. Experimental results are presented in Ch. 5.

4.2 Automatic behavioral Change-Point Detection: A Case Study with Children behaviors

The study of human behaviors in cognitive sciences provides clues to understand and describe people personal and interpersonal functioning. In particular, the analysis of the temporal dynamics and of the interleaving between complex behaviors can be a powerful tool to reveal events, correlations, causalities and synchronous phenomena, but also to discover abnormal behaviors. However, the annotation of these dynamics can be extremely expensive in terms of temporal and human resources demanded to expert annotators.

Therefore, a semi-automated annotation system is proposed to highlight breakpoints and transitions from one behavior to another. Our system is based on a general-purpose methodology that is suitable in the presence of heterogeneous behaviors for which no a-priori model is available.

The methodology has been validated and tested in a semi-ecological use-case scenario: a manually annotated dataset of humans' movements collected during an imitation task of a virtual agent [4, 244] acting as a tightrope walker [245]. The dataset involves behavioral observations from children with typical development (TD) but also children with neuro-developmental disorders (NDD): Autism Spectrum Disorder (ASD) and Developmental Coordination Disorder (DCD). The presence of abnormal movements [246, 247, 248, 249] from children with NDD gives the possibility of validating the proposed system in conditions that are challenging even for experienced annotators.

The behaviors annotated in the tightrope walker dataset are simple and quite stereotyped; however, their effective automated or semi-automated annotation can be generalized and transferred to a larger set of humans' behaviors involving movements. Achieving such effective annotation capabilities would result on fewer human resources committed to the manual analysis of the data, as well as on less time spent on this task.

It is demonstrated:

– The **feasibility** of the proposed approach in practical applications and its **effectiveness** in detecting and classifying change-points in humans' move-

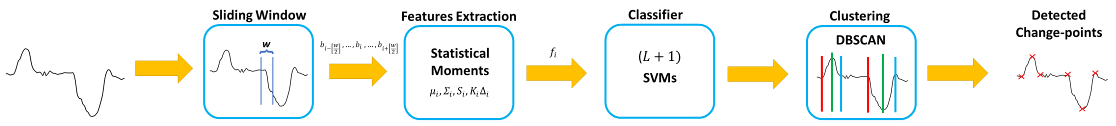


Figure 4.4: The behavioral signal is analyzed in terms of temporal windows. For each temporal window, a description of the statistical moments of the signal within the window are considered. Hence, the descriptor is fed in input to a classifier that returns the predicted change-point class. Predictions for all possible temporal windows are aggregated by a clustering technique. The time instants of the cluster centers are the estimated change-points.

ments;

– The efficacy and the limits of the system while **tackling abnormal behaviors**, for instance when analyzing behaviors of children affected by NDD;

– The impact of the **annotated training set size** to understand how many samples the system training procedure requires to achieve valuable performances.

4.2.1 Mathematical Background

The assumption that the course of specific phenomena, represented in terms of time-series, follows the same fixed stationary process may not be realistic in several domains such as economics, business, engineering, medicine and social sciences. The discovery of specific time points in which the properties of the time-series change is an attracting and well-studied problem [14, 250, 251, 252, 203, 200, 201, 202]. A change-point is defined as the temporal boundary that separates two sequences of observations originating from two different statistical distributions. In this sense, change-point detection can be achieved by analyzing the parameters of such distributions. Changes in the statistical moments of the distribution of the observations can be interpreted as possible change-points. Such changes can be particularly evident in the distribution of characteristics related to the observed measurements such as derivatives or energy spectra. A simple but effective change-point detection method is the CUSUM (cumulative sum) test [253], that involves the calculation of a cumulative sum of the observed characteristics: when the value of the cumulative sum exceeds a certain threshold value, a change-point has been detected.

The change-point detection problem can be stated in terms of a hypothesis test [254, 255] in which the null hypothesis H_0 assumes the absence of a change-point at a specified time in the time-series, while the alternative hypothesis H_1 assumes that there is one. More in the detail, supposing each observation in a time-series originated from some distribution fully described by a set of parameters θ_t with $t \in [0, N)$ indicating time, the hypotheses H_0 and H_1 can be formulated as follows:

$$H_0 : \theta_0 = \cdots = \theta_{p-1} = \theta_p = \theta_{p+1} = \cdots = \theta_{N-1} \tag{4.3}$$

$$H_1 : \theta_0 = \cdots = \theta_{p-1} = \theta_p \neq \theta_{p+1} = \cdots = \theta_{N-1} \tag{4.4}$$

where p represents the time when a change-point event occurs. The key factor in the formulation of H_1 is the inequality $\theta_p \neq \theta_{p+1}$: at some point in the time-series, and precisely between $t = p$ and $t = p + 1$, the underlying distribution changes. Consequently, the hypothesis H_1 assumes the presence of two different distributions characterized by the parameters θ_A and θ_B respectively. When $t \leq p$, $\theta_t = \theta_A$; whenever $t > p$, $\theta_t = \theta_B$.

According to this formulation, it is possible to define λ as the ratio between the likelihoods associated with the two hypotheses H_0 and H_1:

$$\lambda(t) = \frac{\mathcal{L}(H_0|t)}{\mathcal{L}(H_1|t)} \tag{4.5}$$

This likelihood-ratio can provide a general decision test to classify observations in a sequence :

$$\begin{cases} \text{if } \lambda(t) > c & \text{do not reject } H_0 \\ \text{if } \lambda(t) \leq c & \text{reject } H_0 \end{cases} \tag{4.6}$$

where c is a predefined threshold selected to obtain a specified significance level. Without going further into the details, likelihood-ratio based classification systems, such as the Generalized likelihood ratio (GLR) [256] or the marginalized likelihood ratio (MLR) [257], exploit this model by finding suitable parameters that maximize the system capabilities of detecting change-points in the observed measurements.

Non-parametric tests have also been developed for applications in which no prior knowledge about the observed process distribution is available [258, 259]. Other statistical methods use Bayesian prior distributions to incorporate time-dependent information [14, 260].

Without being exhaustive, in recent years many machine learning algorithms have been designed or adapted to the change-point detection problem. Such methods usually employ sliding windows of subsequent observations [261, 262, 252] to reinforce their noise resistance or to capture smoothed transitions that otherwise could be difficult to detect. Under this point of view, supervised approaches have been proposed [261, 262, 263, 264, 265] to model the change-point detection problem as a binary classification problem. In this case, the goal is to discriminate between state transition sequences (i.e., observation sequences including the change-point) and within-state sequences. Multi-class classifiers have been adopted to solve the change-point estimation problem [14] in that case, the goal is the recognition of the type of detected change-point, namely the kind of state transition that arises. Alternatively, unsupervised learning algorithms have been proposed to discover patterns in unlabeled data. Subspace modelling [251, 30] and graph-based technique [266, 267, 268], in particular, have been used as clustering methods to detect change-points.

4.2.2 Methodology

We assume that humans' behaviors are represented through a multi-dimensional time-series $B = \{b_1, b_2, \cdots, b_T\}$ where each sample b_t is a set of observations acquired at time t. We adopt a sliding-window approach, such that W subsequent observations $\{b_i, b_{i+1}, \cdots, b_{i+W-1}\}$ centered in $b_{\lfloor \frac{W+i}{2} \rfloor}$ are used to calculate a behavioral feature $f_{\lfloor \frac{W+i}{2} \rfloor}$. By applying this technique, we transform the time-series B into the time-series $F = \{f_{\lfloor \frac{W}{2} \rfloor}, f_{\lfloor \frac{W}{2} \rfloor + 1}, ... f_{T - \lfloor \frac{W}{2} \rfloor + 1}\}$ where each element f_j is a feature vector extracted from the window centered in the sample b_j.

Provided with this feature representation, we apply a supervised classification technique to detect and recognize change-points from the feature time-series F.

We assume the availability of a dataset annotated by an expert in terms of change-points' timestamp and class, indicating with the latter the behavioral transition to be found. Consequently, our goal will not be the recognition of behaviors but the identification of transitions from one behavior to another. In particular, we consider the problem of jointly recognizing among L different classes of change-points and detecting the time when such change-points arise. We aim at solving this problem without any a priori knowledge of the involved behavior classes. This problem is difficult because: behavior duration and type may largely change, there may be intra- and inter-subjects variations in behaviors and, in general, the transitions from one behavior to another are not abrupt. We adopt a classifier that takes in input the feature descriptor of a temporal window f_t and provides in output the predicted change-point class (a label between 1 and L) or 0 if no change-point has been detected. Since the behavior duration varies, more subsequent windows may be recognized as belonging to the same change-point class. To refine the prediction step and estimate more precisely the instant when the transition arises, we apply a clustering technique to aggregate the predictions. The centroid of each of such clusters is used as estimated change-point.

The framework we implemented is presented in Fig. 4.4. In the following, we provide more details about each of the steps required to apply the above-described methodology.

4.2.3 Feature Extraction

A change-point represents a switch from a dynamical system to another. Hence, in a change-point, the statistical properties of the signal within a temporal window change.

In particular, given a set of subsequent observations $\{b_{t-\lfloor \frac{W}{2} \rfloor}, \cdots, b_t, \cdots, b_{t+\lfloor \frac{W}{2} \rfloor}\}$ in a window of length W, we characterize the signal within the window by its statistical moments. The corresponding feature descriptor is defined as $f_t = \{\mu_t, \Sigma_t, S_t, K_t\, \Delta_t\}$ where: μ_t represents the mean value (1st order moment), Σ_t is the covariance matrix (2nd order moment), S_t and K_t represents the skewness and kurtosis of the multivariate distribution as defined by [269] (3rd and 4th order moments respectively). Moreover, the feature descriptor includes the value Δ_t, which is the difference between the maximal and minimal values of the signal components and, hence, describes the signal extension along with the various components. It is also possible to include the median of the signal values to make the descriptor more robust to outliers.

The feature descriptor can be extended by including also other statistical properties of the signal. It is indeed possible to include the statistical moments of the velocity, acceleration and jerk of the signal (namely, the 1-st, 2-nd and 3-rd order derivatives of the signal) to capture local information about the way the signal changes over time [270, 271, 272].

Besides, the statistical moments of the signal curvature can be included too in the descriptor. The curvature locally measures how fast a curve is changing direction at a given point. The formal definition of curvature is $C = |\frac{d\mathbf{T}}{d\mathbf{s}}|$ where $\mathbf{T}$ is the unit tangent of the curve and $\mathbf{s}$ is the arc length. In particular, for a three-dimensional signal $b(t)$, where t indicates time, the curvature measures the local magnitude of the acceleration of a particle moving with unit speed along the curve and is defined as $C = \frac{||b'(t) \times b''(t)||}{||b'(t)||^3}$, where $b'(t)$ and $b''(t)$ represent 1-st and 2-nd order of time derivatives respectively. For a one-dimensional signal $b(t)$, the curvature is simply defined as $C = \frac{b'(t) - b''(t)}{[1+(b'(t))^2]^{\frac{3}{2}}}$.

To detect and recognize a change-point at time t, and hence a transition from one behavior to another, it may help to also consider *contextual information*, namely the properties of the signal before time $t - \lfloor \frac{W}{2} \rfloor$ and after time $t + \lfloor \frac{W}{2} \rfloor$. As shown in Fig. 4.5, this may be achieved by including in the change-point descriptor f_t the statistical moments of the signal in $\lfloor \frac{W}{2} \rfloor$ observations before and after the t-th window. By doing in this way, the final descriptor f_t has a size equals to $7 \cdot (2k^2 + 3k) + 3k$. In particular, for $k = 1$, the descriptor has a size equals to 38.

The size of the above-described feature vector is independent of the number of considered observations W, namely the size of the temporal window. However, the value of W can have an impact on the descriptiveness of the feature vector. Indeed, the value of W depends on the sampling frequency and the nature of the analyzed signal. Depending on such sampling frequency, a small value of W can be too local and result in a poor descriptor for the change-point detection problem. On the contrary, a too-large value of W increases the risk of using temporal windows that may contain more than one change-point [252]. Our proposed methodology aims at considering the case when no information is available about the duration of each behavior. The selection of the best value for W can be done empirically when an annotated dataset is available by selecting the best classification model among the ones trained by varying the value of W.

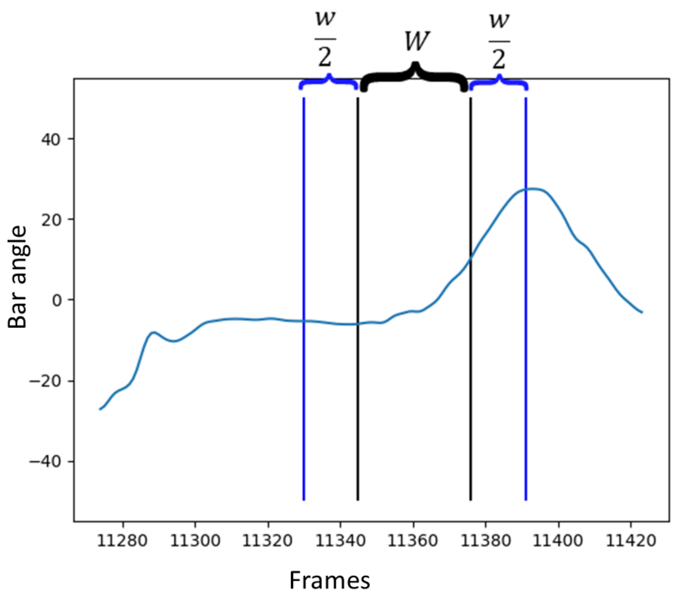


Figure 4.5: Contextual information can be included in the descriptor extracted from a temporal window of length W by computing the statistical moments of the signal of the $\frac{W}{2}$ observations on the left and on the right side of the temporal window itself.

4.2.4 Descriptor Classification

We need to recognize if a descriptor f_j of the j-th window, computed as described in Sec. 4.2.3, is not a change-point or is one of the L possible behavior transitions. We model the problem as a classification one. We assume that a dataset manually annotated by experts in the psychology field is available. In particular, we assume that annotators have provided the time and the class of the change-points in the behavioral time-series. We use the behavioral time-series to extract a set of feature descriptors. We associate to each feature descriptor a label that depends on the available annotations. When considering temporal windows of length W, there are no more than W subsequent temporal windows including an annotated time instant t. Intuitively, the descriptors associated to some of such temporal windows might differ very little (for example, subsequent temporal windows differ for only two observations). Hence, there is the risk to associate two similar descriptors with different labels, which may compromise the classifier training. The most straightforward strategy to avoid this issue would be that of computing a feature descriptor for each window centered in the annotated change-point while avoiding to extract descriptors for overlapping temporal windows. However, this greatly limits the size of the training data, especially for rare change-point classes, with an impact on the accuracy of the classification model. Furthermore, this strategy strongly trusts in the experts' capabilities to precisely annotate change-points. In real experimental scenarios, different experts may not fully agree on the exact location of the change-point in the time-series. In practice, during the data annotation process, different experts annotate independently the data and then discuss and resolve any divergence in the annotation results. Inter-rater reliability is in general measured by the Cohen's Kappa. Such kind of validation increases the cost for annotating data.

This suggests that a safer strategy is that of labelling in a consistent way all feature descriptors extracted from windows centered in observations that are close to the annotated change-point. To this purpose, we set a threshold τ on the absolute difference between the time of the central observation and the annotated change-point.

Another important issue to consider when dealing with change-point classification is the nature of the resulting training set. Indeed, it is not unusual to deal with an unbalanced dataset, namely dataset in which change-point classes are not equally represented. This is especially true for the class representing the within-state sequences [273, 274]. To account for such issue it is important to apply a balancing procedure such that the number of samples in each change-point class becomes comparable.

4.2.5 Change-Point Detection

As detailed in Sec. 4.2.4, our classifier takes in input the feature descriptor of a temporal window and provides a label indicating if the center of the temporal window is a change-point and, in this case, the class of the detected change-point. During test, we apply a sliding window approach, meaning that we classify the feature descriptors of subsequent temporal windows. Two neighbor windows will then differ for two observations and their feature descriptor may not greatly vary. Hence, the classifiers will likely output the same predictions for close temporal windows. We adopt DBSCAN as convenient algorithm to aggregate the time instants of such predictions in compact clusters. Other clustering algorithms need to know the number of clusters they should compute, namely the number of change-points arising in the time-series. In contrast, DBSCAN is an algorithm entirely based on the distances among samples (in our case, time instants); clusters emerge by considering neighborhood relationships among the samples. More in detail, DBSCAN is independently executed on the sets of time instants classified as belonging to a specific change-point class. The average value of the time instants in each resulting cluster is the predicted time when a change-point occurred in the signal.

4.2.6 Case Study

To demonstrate the methodology introduced in this work, we propose its application to the behavioral data collected by [244] and [4] during an imitation task where children had to reproduce the movements of a virtual agent acting as a tightrope walker (TW) [245]. In that scenario, authors focused particularly on how participants were able to imitate the TW and, eventually, perform a perspective change to the point of view of the virtual agent. As we will detail later, children behaviors were described through time-series derived by their hand movements during the imitation task.

In [4], authors explored children behavioral imitation abilities in terms of interpersonal synchronization and motor coordination. In [244], authors investigated the ability of children in performing behavioral own-body-transformations. While in [4] authors exploited the dataset through an automated characterization of temporal slices of the interaction with the virtual agent, in [244] authors analyzed the same dataset by exploiting manually annotated change-points. Notably, in [244], experts have manually annotated change-points in such time-series to analyze and compare the behaviors of TD children and children affected by NDD, specifically ASD and DCD.

In this work, we use the annotated data presented in [244] to apply our methodology and automatically discover change-points in the recorded behavioral signals. We use the data collected with TD children to train and test our system. Then, we use the data collected with children affected by ASD or DCD disorders to further investigate the potentiality and the limits of the proposed methodology. In the Ch. 5 we provided more details about the considered scenario, the data used to asses our method and the implementation details.

Chapter 5

Experimental Results

In this Chapter we discuss the results obtained from the methods presented in the Chapters 3 and 4 of the thesis.

At the beginning of this chapter the experimental results obtained by the trackers introduced in the Secs. 3.2.1 and 3.2.2 is presented. The comparison between these two trackers is possible since the adoption of the same tracking-by-detection paradigm. Later the results of the method based on the Bayesian approach, introduced in Sec. 3.3.1 are shown. The three tracking algorithms were tested on the same public dataset [3], that is also presented here. The trackers of the sections 3.2.1 and 3.2.2 are also compared with a baseline, a CamShift based algorithm, suggested in the work of [3].

To be fair, the presented results are not fully comparable with those in [3] since we validated our methods only on pedestrian targets. The results in [3] consider all the annotated objects of the dataset (including objects), while we focused ourself only on the full-body annotated pedestrian targets. However, also from this point of view our results allowed us to state that our tracking algorithms are accurate and could even be used to track generic objects by using appropriate detectors.

In the following of the Chapter the experimental results on the action recognition framework shown in Sec. 4.1 are presented. Finally, the results of the framework on the human behavior characterization introduced in Sec. 4.2 are discussed.

5.1 Pedestrian Tracking Results

The dataset used to test the tracking algorithms presented in this thesis is the public available dataset of [3]. The video sequences presented in [3] were captured in two indoor environments with five or six randomly moving individuals.

The videos contain a total number of 3.179 panoramic frames recorded at 16 fps. The length of each tracking sequence varies from a few seconds to one/two minutes. These video sequences are affected by common tracking perturbation factors, such as: motion blur, scale change, out-of-plane rotation, fast motion, cluttered background, illumination variation, low resolution, occlusion, presence of distractors and articulated objects. They are also affected by serious illumination artifacts. Moreover there are numerous occlusions arising while people are moving around the camera alone or in groups, which make this dataset challenging.

As baseline method, the work in [3] suggests to use CamShift. In Fig. 5.1 some of the panoramic frames taken from the video sequences of the dataset [3] are shown.

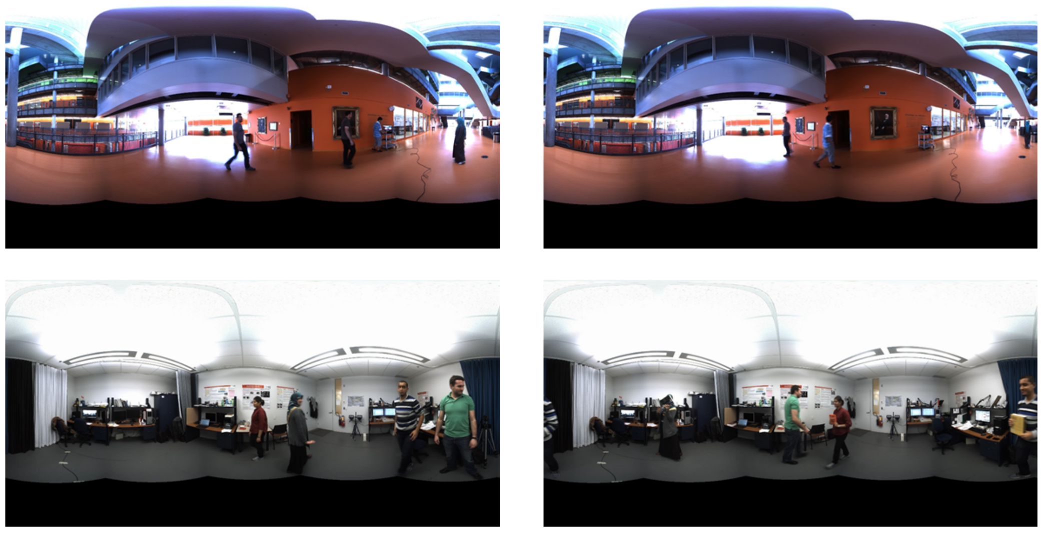


Figure 5.1: Random equirectangular frames taken from the video sequences of [3].

5.1.1 Pedestrian Tracking in 360° video by Virtual PTZ Cameras

The tracker presented in 3.2.1 was tested on the three spherical video sequences of the dataset mentioned above. In [3], the zoom angle of the vPTZ is not predicted and it is always set to 90°. For comparison purposes with [3] even if our algorithm aims at estimating the best zoom angle to closely track the target, the virtual camera resolution of our vPTZ was set to $640X480$, and its field of view θ to 90°.

The tracker was initialized by the first ground-truth bounding box of each target. In the experiments we set $\alpha = 0.5$, and $number\_memories = 9$

In this first method we proposed an approach that allows to track a target in a 360° video by controlling a virtual PTZ camera. The tracking-by-detection approach takes advantage of pedestrian detectors at the state-of-the-art to locate candidate targets in the scene. To select the target among all candidates, this approach considers the appearance similarity of each detection to the target appearance model and the distance of the detection to the center of the image. The appearance model is implemented as a dynamic memory that stores color histograms of past detections. The memory is managed in such a way that only the most discriminative color histograms are retained in memory whenever a new appearance representation is available. Moreover, to decide if updating the memory or not, a threshold on the Bhattacharyya distances is computed online by assuming that distances among detections of the same target follows a Gaussian probability distribution.

To assess our tracker approach, we compare our tracking results to the ground truth by considering two measures: the mean absolute error (MAE) in degrees for the estimated pan (α), tilt (β) and zoom (γ) angles and the Track Fragmentation (TF) error.

The results presented in [3] are obtained by using the *Center Location Error* (CLE), *Overlap Ratio* (OR) and percentage of frames where the tracker successfully followed the target (% of Frames). These metrics are not general enough since these measurements depends on the image resolution of the projected views and on the zoom factor. For the above reason, we decided to not adopt the metrics suggested in [3].

In Table 5.1, we report the values of mean absolute error (MAE) in degrees for the estimated pan (α), tilt (β) and zoom (γ) angles and the values of our modified Track Fragmentation (TF) error.

Method	$MAE(\alpha)$	$MAE(\beta)$	$MAE(\gamma)$	TF
Pf vPTZ [16]	3.14°	2.79°	9.85°	59.84%
CamShift [3]	3.86°	4.68°	36.11°	64.43%
TbyD [15]	2.43°	3.43°	15.26°	70.44%

Table 5.1: Experimental Results: MAE stands for Mean Absolute Error, α, β, γ indicate pan, tilt and zoom respectively, TF stands for Track Fragmentation

As shown in Table 5.1, our method outperforms the CamShift algorithm in terms of MAE and TF and, in particular, it seems to be able to handle better the zoom factor estimation. This experimental results showed that overall the idea of using virtual PTZ cameras to perform tracking in 360° videos is viable. They have also shown the limits of the simplistic adopted appearance model.

5.1.2 Tracking-by-Detection in 360° videos using pre-trained deep models

In the tracker presented in 3.2.2 efforts have been made to make the target appearance model more robust. For this reason, in this tracking algorithm the segmentation masks estimated by Mask R-CNN were used to compute a simple but robust target appearance model based on Hue-Saturation-Value (HSV) color histograms. The Mask R-CNN is based on the ResNet101 backbone with the pre-trained weights of the MS COCO dataset [275]. The min and max dimensions of the width and height of the input images for the Mask R-CNN were set both to 128. This size allows the network to perform detection on images very quickly. The min and max dimensions of input images should not be too small cause some detections could be lost during the rescaling performed by the model of the network. Mask R-CNN was set with the minimum probability value to accept a detected instance (ROIs below this threshold are skipped), equal to 0.7 and with the non-maximum suppression threshold for detection, important to get rid off overlapping prediction masks, equal to 0.3. In our experiments, we tried various configurations for the Kalman filter state space, as shown in table 5.2. The progressive zoom out strategy performed during a missing or occlusion was set to a $\lambda = 10°$, and increased frame by frame of the same amount, until the occlusion or missing is present, up to a maximum value of 30° .

We noted that the correction of tilt and zoom parameters with the Kalman filter modifies the camera FOV in such a way that it does not allow the Mask R-CNN to re-detect the target in the next frames. The continuous correction of the tilt leads the vPTZ to move either too high or too low in relation to where the target is actually located. This causes that the Mask R-CNN does not detect the target after few frames. Even the zoom correction causes an analogous situation, because of an exaggerated zooming-in.

According to this, we decided to apply the correction and prediction of the only pan to the Kalman filter. In all our experiments the threshold μ for the target association and the α parameter was both empirically set to 0.10. In our experiments we have observed that small variations of these two parameters do not have much influence on the accuracy of the method. For comparison purposes, we adopted the same evaluation protocol as in [15].

In our experiments, we tested several configuration by varying the number of color histograms to model the target appearance and the vPTZ camera parameters to correct by the Kalman filter.

The $2H - Kalman(\alpha, \beta, \gamma)$ and $2H - Kalman(\alpha)$ configurations divided the target image into upper and lower parts to compute two different histograms in the HSV color space, as presented in [15, 16].

In the $2H - Kalman(\alpha, \beta, \gamma)$, Kalman filter was adopted for correcting and predicting the pan, tilt and zoom for the vPTZ camera in case of occlusions or missing detection. In $2H - Kalman(\alpha)$, the Kalman filter is used for correcting and predicting only the pan parameter.

The $1H$ and $1H - Kalman(\alpha)$ use the entire target image to compute the color histogram (without splitting it in two parts). The $1H$ configuration does not use any Kalman filter and performs tracking by linking over time the detections provided by the Mask R-CNN. The $1H - Kalman(\alpha)$, instead, also adopts the Kalman Filter to correct the pan parameter.

In Table 5.2, we reported the performances of our method with the same metrics proposed in [16] and [15]. In particular, we reported the values of mean absolute errors (MAE) in degrees for the predicted pan (α), tilt (β) and zoom (γ) angles of the various configurations. Furthermore, we reported the Track Fragmentation (TF) error that measures the percentage of invalid detections in the tracks.

As we can see in the table 5.2 the configurations with the full histogram returned better results than those with the splitted one and all our configurations obtained better results than [15], overcoming the state-of-the-art for the [3] dataset. In particular, we can see how the TF is low in the $1H - Kalman(\alpha)$ configuration. The use of the kalman filter on all the three pan, tilt and zoom parameters of the vPTZ camera was found to be disadvantageous, as can be seen from the TF of the $2H - Kalman(\alpha, \beta, \gamma)$ configuration.

We also tried a configuration that included the Kalman filter for the prediction of the three parameters pan, tilt and zoom both during occlusion/missing and non-occlusion/missing situations. We decided to not report this configuration because the tracker after a few frames was not able to follow the target anymore. This is due to the fact that the corrections of the pan, tilt and zoom of the Kalman filter returned values that do not allow the Mask R-CNN to properly detect objects.

Configuration	$MAE(\alpha)$	$MAE(\beta)$	$MAE(\gamma)$	TF
$2H - Kalman(\alpha, \beta, \gamma)$	2.81°	3.95°	10.41°	36.1%
$2H - Kalman(\alpha)$	2.10°	1.32°	9.22°	22.1%
$1H - NoKalman$	**1.13°**	1.45°	9.15°	22.5%
$1H - Kalman(\alpha)$	1.95°	**1.25°**	**8.87°**	**12.6%**
TbyD [15]	2.43°	3.43°	15.26°	70.44%
Pf vPTZ [16]	3.14°	2.79°	9.85°	59.8%

Table 5.2: Experimental Results: MAE stands for Mean Absolute Error, α, β, γ indicate pan, tilt and zoom respectively, TF stands for Track Fragmentation

5.1.3 Particle Filtering for Tracking in 360° videos using virtual PTZ Cameras

In the Sec. 3.3.1 we introduced the idea of using particle filter with vPTZ cameras. In this approach we took advantage of the particle filter to model the posterior distribution of a vPTZ camera views. The tracker was initialized by the first bounding box of the target provided with the ground-truth. To initialize the tracker automatically, a pedestrian detector could be used on the vPTZ camera views obtained by setting the zoom angle to 90°, the tilt angle to 0°, and by varying the pan angle in $[-180, -90, 0, 90]$.

As shown in Table 5.1, our method outperforms both the CamShift algorithm and our previous method [15] in terms of MAE and TF and, in particular, it seems to be able to handle better the zoom factor estimation. All methods show high values of TF, whilst the proposed method achieves the lowest one. We note here that all methods perform tracking in the HSV color space by employing color histograms. Overall, the results suggest that by adopting stronger appearance descriptors the performance of the method could improve.

In the proposed framework, the posterior distribution is approximated by a discrete set of particles. Each particle represents a vPTZ camera with specific values of pan, tilt and zoom. Such cameras are weighted and used, frame-by-frame, to estimate the vPTZ camera to track the target. At the best of our knowledge, there are no works using the particle filter framework to model the posterior distribution of vPTZ cameras.

Adjustments must be made to improve the observation likelihood model and, in general, the appearance model used to describe the target in order to make it adaptive. Targets have been tracked over all the video frames. An automatic procedure that ends the tracker when the target is not detected for a prefixed number of consecutive frames could also be introduced. We will investigate the possibility to extends our tracking approaches to multiple targets. For example, a tracker could be initialized for each target and information about the estimated targets' locations on the sphere could be shared among trackers in order to disambiguate among them.

5.2 Human Behavior Characterization Results

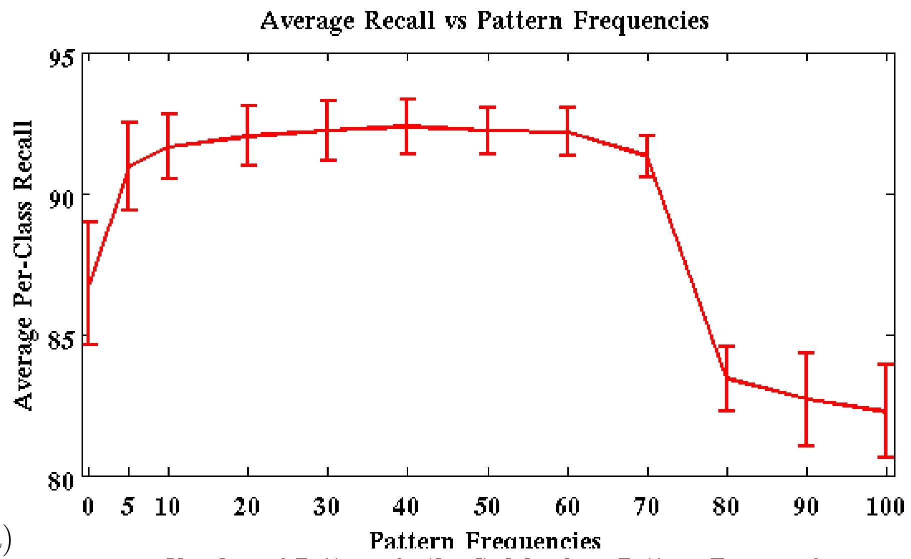
In this section, the experimental results of the action recognition framework shown in Sec. 4.1 and of the human behavior characterization framework introduced in Sec. 4.2 are presented.

5.2.1 Hop: Histogram of Patterns for Human Action Representation

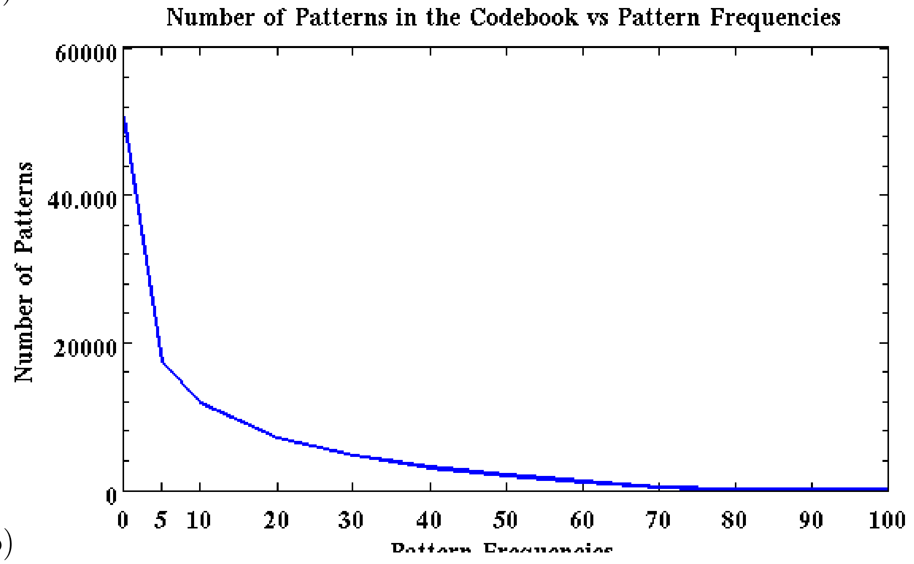
We validated the framework of *Hop: Histogram of Patterns for Human Action Representation* on the Microsoft Research Cambridge-12 (MSRC-12) gesture dataset [242]. The dataset consists of sequences of skeletons described by means of the coordinates of 20 3D body joints. Skeletons were estimated by using the Kinect Pose Estimation pipeline [137]. The dataset includes 594 sequences representing the performances of 12 actions (*Start system* (SS), *Duck* (D), *Push Right* (PR), *Goggles* (G), *Wind it up* (W), *Shoot* (S), *Bow* (B), *Throw* (T), *Had enough* (H), *Change weapon* (C), *Beat both* (BB), *Kick* (K)) from 30 different subjects. Each sequence is a recording of one subject performing one gesture several times. Considering that the MSRC-12 dataset has been proposed for action detection, no temporal segmentation of the single performance is provided with the dataset but only the time when the action is considered recognizable. In order to test our method in action classification, we adopted the annotation made publicly available by [276]. Such annotation specifies the initial/final frame when each performance starts/ends. This annotation has produced 6243 different action sequences. In order to account for biometric differences, we preprocessed each action sequence by removing its average skeleton. In general, mining algorithms are used over a single sequence to discover repetitive patterns. In contrast, our algorithm learns frequent patterns over the entire training dataset, which includes segmented action sequences from different classes and performed by different subjects. Thus, our training approach allows us to learn more general frequent patterns. We repeated the experiment 10 times in cross-validation with a 50% subject split experimental protocol, that is we randomly select half of the subjects to build the training set, while the sequences of the remaining subjects are used for test.

The training set has been used to learn a codebook of frequent sequential patterns, and to train one vs one χ^2 kernel SVMs with C equals to 10. In our modified apriori algorithm we empirically set minimal and maximal pattern length respectively to $\tau = 3$ and $K_M = 30$. The similarity threshold ϵ used to establish a match between pattern candidates and time windows was set to 0.9, while the threshold ψ was set to 75.

T vs P	SS	D	PR	G	W	S	B	T	H	C	BB	K
SS	**80.23**	0	0	0	0.43	4.18	0.04	0.90	4.59	1.07	7.63	0.93
D	0	**99.96**	0	0	0	0	0.04	0	0	0	0	0
PR	0.04	0	**96.35**	0	0.73	1.42	0	0.24	0.12	1.09	0	0
G	0.12	0	0	**93.14**	1.00	1.66	0	0	3.12	0.48	0.48	0
W	0.42	0	1.24	0.09	**92.43**	1.18	0	0.10	0	2.00	2.54	0
S	0.59	0	0.07	0.11	0.30	**93.76**	0.04	0.04	0.12	2.28	2.67	0
B	0	4.38	0	0	0	0.20	**95.15**	0.04	0	0.03	0	0.19
T	0.04	0	0.08	0	0.04	0.81	0.44	**93.10**	0	1.42	0.04	4.03
H	2.74	0	0.04	5.35	0.12	1.28	0	0	**89.00**	0.04	1.42	0
C	0.08	0	0.04	0.08	0.28	3.31	0	0	0.04	**95.77**	0.40	0
BB	3.19	0.62	0.08	0.41	3.89	6.22	0	0.15	1.61	2.47	**81.33**	0.04
K	0.20	0.07	0	0	0.04	0.24	0.30	0.35	0	0.49	0	**98.30**



a)



b)

Figure 5.2: Plots in a) and b) show how the average per-class recall and the number of patterns in the codebook, respectively, change by varying the minimal pattern frequency. Values are averaged over 10 runs, and vertical bars show standard deviations.

We performed experiments to test the quality of the codebook of frequent sequential patterns generated via Alg. 2. On average, our codebook has a size of 120 ± 14.72 patterns. The average accuracy value over 10 runs is approximately of about 88.32%. This result is very encouraging considering that the action representation is very compact.

As detailed in Section 4.1.2, the codebook stores all patterns with a frequency count greater than ψ that do not contribute to the generation of longer patterns. However, since we adopt an approximate matching strategy, the frequency count of the generated patterns is not a very reliable measure of the importance of the learned patterns. Hence, it is reasonable to wonder if patterns that are considered infrequent during the codebook learning procedure might actually improve action classification. To validate our hypothesis, we also included in the codebook sequential patterns that are pruned in line 11 of Alg. 2 and having a frequency count greater than a threshold ϕ. Then, we study how frequent a frequent pattern should be for being included in the codebook by studying how the average recall changes when varying ϕ in the range $[0, 100]$.

Figure 5.2.a) shows the trend of the average per-class recall over 10 runs when varying ϕ. Vertical bars represent standard deviations of recall values. Figure 5.2.b) shows the number of patterns in the codebook with a frequency greater than ϕ. As shown in the latter plot, the codebook size decreases exponentially; on average, the codebook size ranges between 50583 (when $\phi = 0$, i.e. all infrequent patterns are included in the codebook) and 44 (when $\phi = 100$).

On the other hand, as shown in figure 5.2.a), there is an increase of the recall values for growing values of ϕ. For value of ϕ in $[20 - 70]$ there is a very limited variation of the average recall; what it really changes is the codebook size that affects the complexity of the vector quantization step. The best average per-class recall is obtained for $\phi = 40$ and is of about $92.38\% \pm 0.97$. The corresponding codebook size is of about 3086. For $\phi = 70$, the average recall is of about 91.31% and the codebook size is on average 400. For $\phi > 70$, the recall value decreases, however the information embedded in very frequent patterns is still very high considering that, with only 44 codewords (on average) with $\phi = 100$, the method achieves an average recall of about 82.26%.

Table 5.2.1 shows the confusion matrix obtained with our technique averaged over 10 runs when $\phi = 40$. Columns of the table represents predicted class labels while rows represent true class labels. As shown in the table, most of the confusion is between the action classes *Start System (SS)* and *Beat both (BB)*, *Had enough (H)* and *Goggles(G)*, *Beat both (BB)* and *Shoot (S)*. We stress here that this technique has been tested directly on the 3D joints coordinates and the only preprocessing of the sequences consists of making them zero mean. Since the method is very general, we believe that the use of more complex features extracted from skeletal data might result in higher value of the average recall.

We compare our method against the work in [276] on equal terms of experimental protocol. In [276], a pyramid of covariance matrices of 3D joints coordinates is used to represent a sequence of skeletons: the root node encode information about the entire sequence; at lower levels, sequences of covariance matrices calculated by a sliding window approach are considered. Action classification is performed by linear SVM. The framework only reports the average correct classification rate or accuracy value averaged over 10 runs in different configuration, and achieves the best accuracy value of about 91.7%. Our accuracy value is of about 92.31% at $\phi = 40$, which is slightly superior to the one of [276].

So, in this framework we demonstrate the idea of representing sequences of skeletons by means of distributions of frequent patterns. In our framework, frequent sequential patterns are computed by means of a modified version of the apriori algorithm. At each iteration, all frequent patterns that cannot be used for generating longer patterns are stored and used as codewords. This approach yields to a codebook of patterns of different length.

To encode the data, at each frame, we use a temporal window whose length adapts to the length of the pattern. Then, the most similar pattern is found and the histogram is updated accordingly.

5.2.2 Automatic behavioral Change-Point Detection: A Case Study with Children behaviors

To validate the system presented in Ch.4, we conducted a series of experiments on the tightrope walker database that we are going to present below. In particular, in this framework we focused on:

– **The evaluation of the performances of the system** in detecting and recognizing change-points in behaviors of TD children. Performance metrics computed in this case can be seen as measures from a *best-case scenario* since the signals representing children's behaviors are smooth and readable (see Fig. 5.6). The method was compared against a baseline method.

– **The efficacy and the limits of the system** in presence of *NDD*. In this particular case, the system deals with abnormal signals that are not easy to be labeled even for expert annotators (see Figs. 5.4 and 5.5);

– **The trade-off** between training set size and achieved system performances.

To assess the system capability of precisely detecting change-points, we measured the F1-Score, the Mean Absolute Error (MAE), the Missing Rate (MR) and the Precision (P) achieved by our system.

- The F1-score is computed for the (binary) detection problem. In this way, all feature vectors that are classified as belonging to one of the change-point classes are considered as positive cases. The vectors that are classified as within-state sequences are considered as negative cases. The F1-score is defined as:

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{5.1}$$

  where P indicates the precision value and R stands for recall, and measures the proportion of true positives with respect to the number of annotated positive samples.

- The Mean Absolute Error (MAE) measures the average absolute difference between the time instant when a change-point has been detected and the time instant when the change-point has been annotated. Time is expressed as frame number (data have been collected with a frame rate equals to 25). MAE is computed only for the valid detection. A detected change-point is considered valid if its distance to the annotated change-point in terms of number of frames is lower than 100.

- The Missing Rate (MR) is defined as the percentage of annotated change-points that are not detected by our system.

- The precision (P) is defined as the proportion of true positives with respect to all positive predictions made by our system.

Experimental protocol

Authors of [245] used a colour movie of a computer-generated female tightrope walker (TW) to investigate whether individuals, under spontaneous conditions and without explicit instruction, embody another person's behavior. They designed a motor paradigm focusing on elementary mimicry to investigate, from the body posture, how individuals act together, focusing in particular on the achievement of own-body transformations, from an embodied, ego-centered viewpoint to a disembodied, hetero-centered viewpoint.

The studies from [4] and [244] extended the paradigm of the TW to adapt it to the study of children behaviors by adjusting the size of the virtual character, giving it a cartoon-like aspect of a child. Fig. 5.3 shows an image of the TW in the front-facing orientation: the artificial TW projected on the wall stands on a rope and keeps its balancing by carrying a bar that, over time, it tilts laterally to its right or left. The TW's bar tilts have a maximum amplitude of 51° and a maximum duration of $3.2s$ (mean duration: $2.7s$). Tilts on the right or left are performed in random order 7 times. Children were provided with a wooden bar and invited to mimic the TW's behaviors by rotating their bar accordingly. To elicit perspective taking, two kinds of sequences were alternated 7 times: in the first type of sequence, the TW walks after the participant, by back-facing her/him; in the second type, the TW walks towards the participant, by front-facing her/him. In the first case, the participant can simply imitate the TW; in the second case, an effective imitation from the participant implies a mental rotation from the point of view of the TW.

Authors of [4] and [244] found behavioral differences in these tasks between TD children and ASD, DCD children. Hence, we expect that the change-point detection and recognition in behavior time series of ASD and DCD subjects will be harder than in TD subjects considering the behavioral differences between these subject groups [277, 278]. In particular, we suppose the system will have more difficulties in dealing with ASD than DCD subjects.

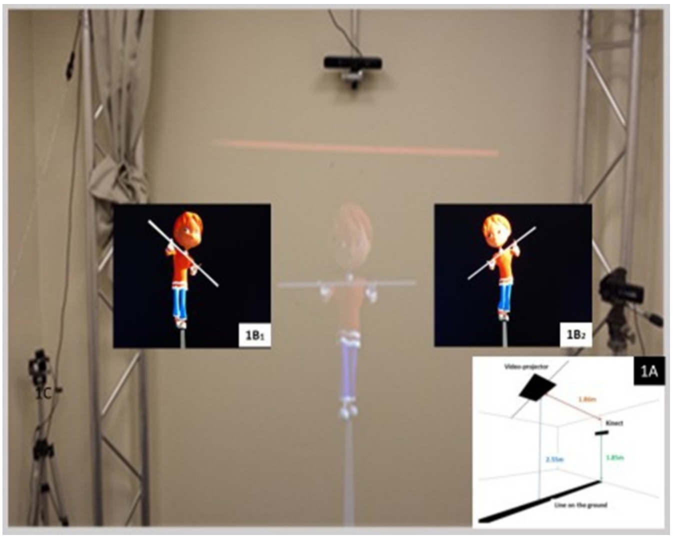


Figure 5.3: The image shows the real-life projected Tightrope walker used in [4]. Images on the left and on the right show the TW while tilting its bar to the left and to the right.

Dataset

During each experimental session, videos of the participants were automatically and continuously recorded for offline analysis by a RGB-D sensor located in front of them, a Microsoft Kinect[1]. This sensor was used to estimate 3D poses of the participant (in terms of 3D skeletal data) at a frame-rate of about 25fps. The angle of the child's bar tilt is calculated using the inclination of the 3D line passing through the two 3D points representing the child's hands.

The dataset includes experiments from 85 children imitating the TW. According to the experimental protocol, each experiment is composed by 7 sequences; however, only 70 children had completed all the planned sessions. Of these 70 children, 30 are TD, 14 DCD and 26 ASD.

Overall, the dataset adopted in this framework includes 490 bar tilts sequences of which 210 are of TD children, 98 of DCD children and 182 of ASD children.

Fig. 5.6 shows the signals measured during an experimental session. In particular, it shows the measured tilt angles in degrees of the bar of one of the participants (the blue line) and of the TW (the orange line) over time. As shown in the figure, the child's bar is initially in a horizontal position where the tilt angle is equal to 0°. Then, for 7 times, the bar is moved and the tilt angle increases (decreases) till reaching a peak value. Later on, the tilt angle decreases (increases) till reaching again the value 0 (bar in horizontal position).

Change-points in [244] were manually identified by expert annotators by considering the starting time of the bar movement, the time when the tilt angle reaches its peak value, and the time of the bar end movement (see supplementary materials in [244]). Hence, three different change-point classes have been annotated. Despite each sequence is composed by 7 tilt movements, experts found that only 6 of them were meaningful and reliable enough to be used for the analysis of the child's imitation capabilities. Therefore, each sequence of angles is annotated with 18 change-points (6 for each change-point class).

Annotating change-points in the signals acquired with ASD and DCD children is a complex activity even for psychiatrists [4]. Figs. 5.4 and 5.5 show examples of the tilt angle signal measured with a DCD and a ASD child respectively. As shown in the figures, the signals are less smoothed than the ones acquired with TD children. This is probably due to the difficulty of these children in replicating/imitating the TW's movements.

[1]Microsoft Kinect website: https://developer.microsoft.com/en-us/windows/kinect

Overall, for the 3 different classes of change-points, the data include 8820 annotated change-points in the tilt angle sequences collected with all the 70 children. In particular, 3780 of these change-points are annotated in sequences collected with TD children.

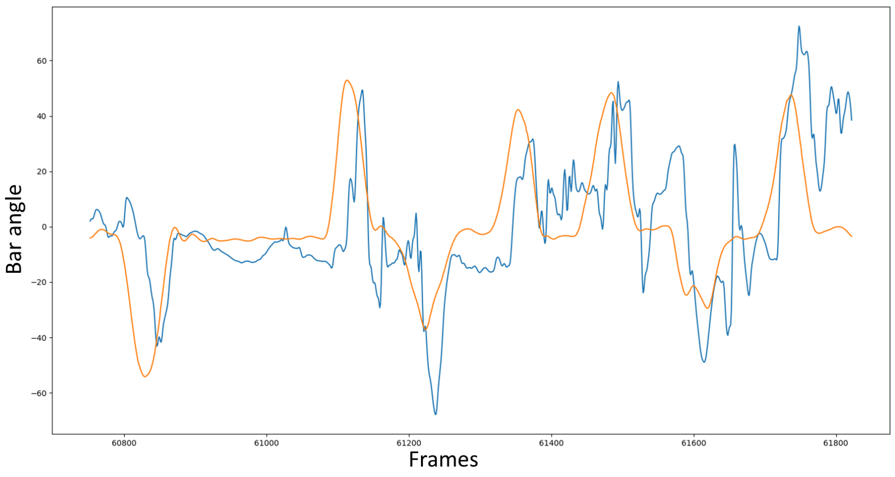


Figure 5.4: The blue line represents the tilt angle of a DCD child's bar. The orange line represents the reference tilt angle of the TW's bar.

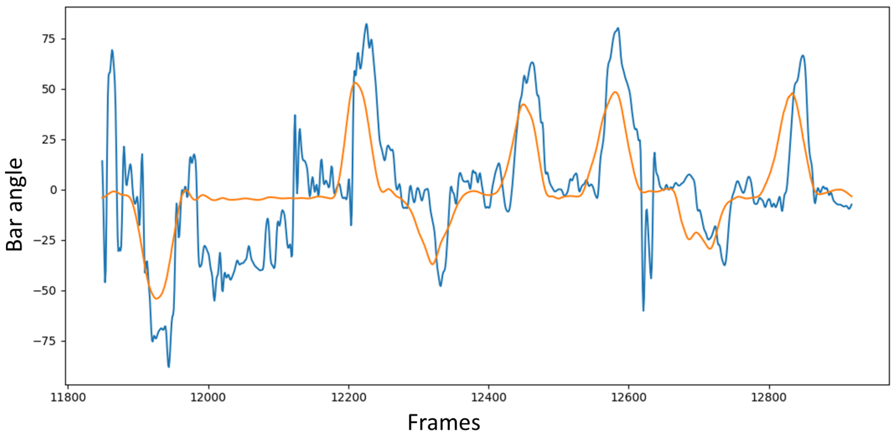


Figure 5.5: The blue line represents the tilt angle of a ASD child's bar. The orange line represents the reference tilt angle of the TW's bar.

Implementation details

Our system has been trained considering the 3780 change-points annotated in the TD children data. Around each of these change-points, by setting $\tau = 2$, five temporal windows have been built by setting $W = 31$. Hence, for each change-point class, 6300 feature descriptors have been extracted. To maintain the dataset balanced, 6300 feature descriptors have been extracted from randomly selected temporal windows representing within-state sequences. We trained SVMs with a 1-vs-1 strategy for the 3 change-point classes and the additional within-state, background class by using RBF kernel.

Before extracting the feature descriptors, we applied a smoothing filter on each sequence and, in particular, we used the Savitzky-Golay filter.

Before training the classifiers, z-score normalization was applied to the data.

During test, we adopted a sliding window approach and classified all windows with a stride equal to 1. Then we aggregated the positive predictions by applying the DBSCAN algorithm. The minimum number of samples in each cluster was set to 3. All experiments were performed in cross-subject validation.

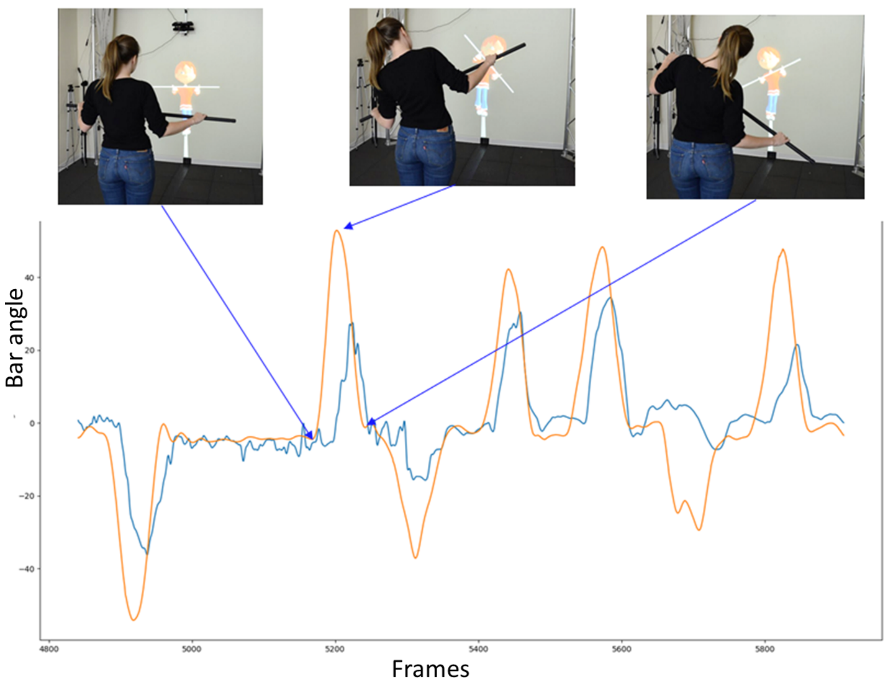


Figure 5.6: The blue line represents the tilt angle of a TD child's bar. The orange line represents the reference tilt angle of the TW's bar.

Evaluation of the performances of the system

A leave-one-subject-out protocol has been employed to validate the presented system using sequences collected from TD children: for each subject, we trained a model with the samples from all the other subjects; samples from the excluded subject were used as test set. The protocol has been repeated using two different window lengths ($W = 31$

and $W = 61$) and two different descriptors: a simple one ($S - SW$) resuming the statistical properties of the signal in each window, and an extended one ($C - SW$) able to include contextual information considering $\lfloor \frac{W}{2} \rfloor = 15$ observations before and after each temporal window, as discussed in 4.2.3. Average performances were compared between them and against a baseline method.

The baseline method computes the second order derivative of the behavioral signal to measure the concavity of the signal itself. The second order derivative is then filtered by discarding all values below a threshold σ. The threshold σ is set to the standard deviation of the second derivative in the training data. The time instants of the unsuppressed values are clustered by using the K-means algorithm. The value K is set to the number of annotated change-points in the analyzed sequence, which is 6 for each change-point class in each experimental session as detailed in Sec.5.2.2. The centers of the found clusters are considered as predicted change-points. By using a-priori knowledge of the order in which the behavioral classes appear in the sequence, each detected change-point is associated with a category. We note here that the baseline method is simpler than the method we propose but is not fully automatic because it requires a-priori knowledge of the number of expected change-points and of the order in which behaviors emerge over time.

Table 5.3 shows the results we obtained. For each metric, computed as average over the per-subject predictions, we report the average value and the standard deviation.

Metric:	F1-score	MR(%)	MAE	P
C-SW ($W = 31$)	**0.95** ± 0.05	**3.4** ± 5	**4.07** ± 1.64	0.938 ± 0.069
S-SW ($W = 31$)	0.94 ± 0.05	3.7 ± 5.5	4.31 ± 2	0.933 ± 0.075
S-SW ($W = 61$)	**0.95** ± 0.04	4 ± 4.9	4.65 ± 1.71	**0.95** ± 0.058
Baseline M.	0.71	14	9.38	0.62

Table 5.3: The table reports the performance of our system (C-SW) on sequences collected from Typical Children. S-SW does not use any context information. In bold font, we report the best value obtained for each metric.

Table 5.3 shows that all variants of the proposed method (C-SW and S-SW) outperform the baseline method in all the selected performance metrics. The table also shows the impact of the window size W on the system performances: while the F1-Scores and the precision (P) do not change much, the amount of missing rates (MR) and the mean average error (MAE) slightly increase when enlarging the window size suggesting that the value of W can affect the general reliability of the system in terms of accuracy in localizing the change-points. Consequently, a meaningful choice of the window length would be a compromise of 31 frames ($W = 31$). Since the system captures data at 25 fps, a value of W equals to 31 turns out to process temporal windows slightly larger than 1 second. Finally, the table shows that embedding contextual information into the descriptor ($C - SW$) tends to improve the system performances.

As discussed in Sec. 4.2.4, to train our model it is necessary to associate each feature vector with a label indicating whether a change-point occurs in the considered window and, eventually, the type of change-point. This labeling task can be a challenging activity that can be performed through different strategies.

It is possible to label as change-point only the window in which the central frame has been annotated as change-point; otherwise, it is possible to annotate as change-point all the windows such that the distance between the annotated change-point and the central observation is below a threshold τ.

Experiments presented in Table 5.3 have been performed with the latter strategy by setting $\tau = 2$. Table 5.4 compares such results with those obtained with the former strategy ($\tau = 0$). As shown in the table, the performance obtained by setting $\tau = 0$ are slightly inferior to those obtained by setting $\tau = 2$.

Metric:	F1-score	MR %	MAE	P
C-SW(31, 2)	**0.95** ± 0.05	**3.4** ± 5	**4.07** ± 1.64	0.938 ± 0.069
C-SW(31, 0)	0.946 ± 0.05	4.3 ± 6.5	4.40 ± 1.79	**0.94** ± 0.065
S-SW(31, 0)	0.94 ± 0.05	4.0 ± 6.7	4.39 ± 1.55	0.93 ± 0.072
S-SW(61, 0)	0.948 ± 0.04	4.1 ± 4.8	4.65 ± 1.71	**0.94** ± 0.058

Table 5.4: The table reports the performance of our system (C-SW) with $\tau = 2$ and $\tau = 0$ on sequences collected from Typical Children. The first number indicates the value of the variable W.

Finally, we analysed the change-point class recognition capabilities of the system. Table 5.5 reports the confusion matrix for the change-point classes "Start", "Peak", "End" and "Background". In particular, the Background class represents the within-state class. The confusion matrix highlights that the system is able to correctly detect and recognize change-point classes with an interesting degree of reliability.

T vs. P (TD)	Start	Peak	End	Background
Start	94.96%	0.083%	0.16%	4.78%
Peak	0.25%	93.3%	1.09%	5.36%
End	0.16%	0.13%	92.2%	7.51%
Background	0.024%	0.035%	0.057%	99.88%

Table 5.5: The table shows the confusion matrix for the change-point classification problem on behavioral data from children in typical development.

Performances in presence of abnormal behaviors

We investigated how the system deals with data coming from abnormal behaviors. Therefore, from the tightrope walker database we selected the data of children with NDD. As already pointed out in Sec.4.2.6, annotating such kinds of behavioral signals is challenging even for experts, hence, any system, automated or semi-automated, able to help and speed-up this process is especially important. In any case, due to such difficulties, we expect a drop in the performance of our system.

Average performances of the system where obtained by training it with TD data only, using the most effective parameter set previously found. Then, the obtained model has been tested with data from children with NDD. As consequence, we asked to the system an important ability of generalization and abstraction from the training sample set.

Metric:	F1-score	MR %	MAE	P
TD	0.95 ± 0.05	3.4 ± 5	4.07 ± 1.64	0.938 ± 0.069
DCD	0.91 ± 0.001	4.2 ± 0.26	4.38 ± 0.15	0.88 ± 0.002
ASD	0.82 ± 0.002	6.5 ± 3	5.2 ± 0.14	0.73 ± 0.03

Table 5.6: The table reports the performance of our system ($C - SW(W = 31)$) on sequences collected from three groups of children: TD, ASD and DCD.

Table 5.6 compares the performance achieved by our system ($C - SW(W = 31)$) for the three groups of children: TD, DCD and ASD. More in detail, we conducted an analysis of the recognition performances of the system in presence of NDDs. Table 5.7 and 5.8 report the confusion matrices on the change-point classes predicted by the classifier on behavioral data from children with DCD and ASD, respectively. Such confusion matrices highlight that the system is still able of distinguishing the change-points from the background class, whilst the recognition performances of the change-point classes deteriorate, especially when analysing ASD children data.

T vs. P (DCD)	Start	Peak	End	Background
Start	92.5%	0.21%	0.21%	7%
Peak	0.0%	92.5%	0.63%	6.77%
End	0.21%	0.21%	86%	13.34%
Background	0.034%	0.033%	0.065%	99.86%

Table 5.7: The table shows the confusion matrix on the predicted change-point classes when analysing behavioral data from children with developmental coordination disorder (DCD).

T vs. P (ASD)	Start	Peak	End	Background
Start	88%	0.38%	0.25%	10.77%
Peak	0.76%	86%	0.12%	12.54%
End	0.88%	0.63%	83%	14%
Background	0.044%	0.051%	0.060%	99%

Table 5.8: The table shows the confusion matrix on the predicted change-point classes when analysing behavioral data from children with autism spectrum disorder (ASD).

Overall, as highlighted also by [4], the system performances decay on NDD children data, highlighting the difficulties the system has in analysing such data. Notably, the performances deterioration show a correspondence with the examined pathological groups: since DCD is a deficit on fine motor control skills, observations from DCD children are less precise than the ones collected from TD children because anomalous movements are introduced; in ASD children performances got worst as effect of motor control and social interaction deficits, that manifest themselves in more abnormal and erratic movements.

Trade-off between training set size and system performances

Finally, we have investigated how the size of the training set affects the performance of our model. Using a cross-subject validation protocol, we have trained our classifier with a variable number of subjects

varying within $\{5, 10, \ldots, 95\}$% of the total number of TD children. As shown in Fig. 5.7, by varying the percentage of subjects in the training set, the MAE value decreases. This result clearly shows that a wider training set allows for a more precise localization of the change-points. In a similar way, the MR value also decreases. Finally, while the F1-score keeps growing while increasing the size of the training set, the precision value has an inflection and starts to grow again after the 80% of the subjects are used for training purposes. Overall, with small dataset size, the system exhibits high recall but is not precise yet. By increasing the size of the training set, the precision value increases as well.

Notably, the experimental results highlight how the system needs just the 30% of the total number of subjects to achieve a 90% of the F1-score. Consequently, this methodology will result in a reduction of the 70% of the effort spent on the annotation of the whole dataset.

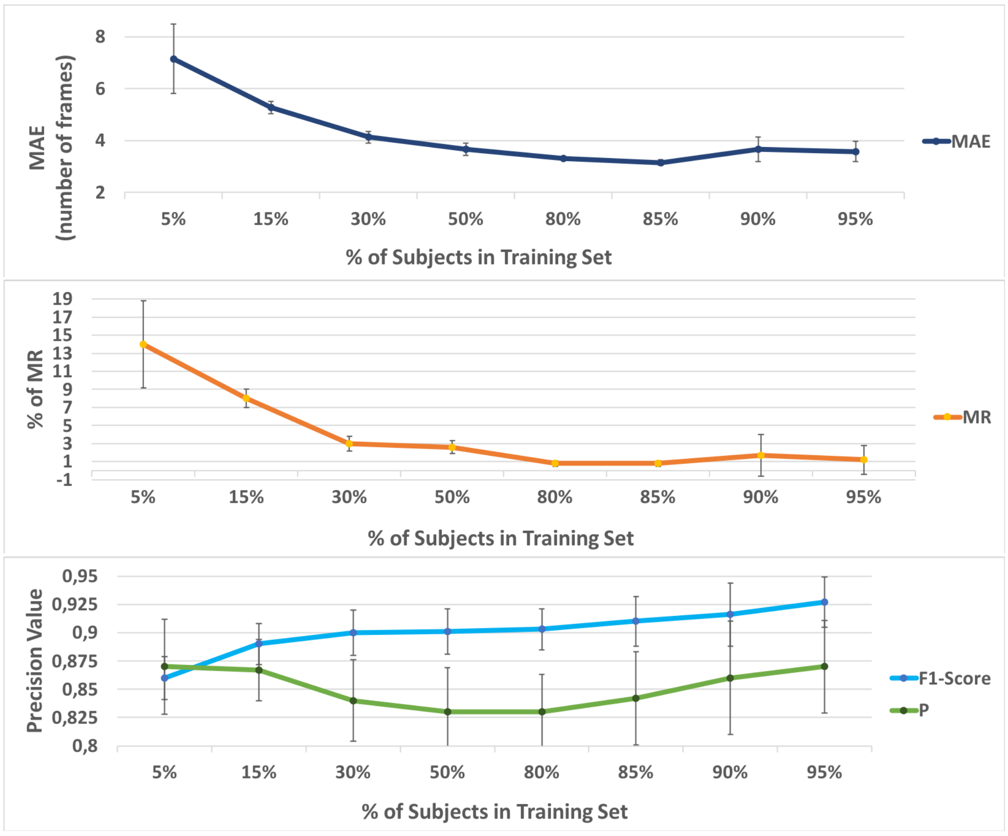


Figure 5.7: The plots are obtained by varying the percentage of subjects in the training set. The first plot shows the trend of the MAE, the second one shows the values of the MR and, finally, the third plot represents the F1-score and the Precision values.

Chapter 6

Conclusions and Future Work

In this thesis some of the problems and difficulties related to human motion analysis were discussed. Human motion was categorized into human motion tracking, human activity recognition and motion analysis of human body parts. These topics were discussed in Chs. 3 and 4.

6.1 Pedestrian Tracking

We presented a framework to simulate PTZ cameras (vPTZ) based on 360° videos. This framework was used to compare tracking algorithms for PTZ cameras. Indeed, one of the main challenges in PTZ camera-based tracking is that results are not easy to reproduce. This explains the lack of standard evaluation protocols in this field. By simulating PTZ cameras from 360° videos, this framework allowed not only the direct comparison of PTZ tracking algorithms but also to investigate the limits of PTZ tracking approaches. Different approaches to track a target in a 360° video by controlling a vPTZ camera were proposed. Two of these rely on tracking-by-detection approach while another one on a Bayesian framework. Tracking-by-detection takes advantage of pedestrian detectors at the state-of-the-art to locate candidate targets in the scene. To select the target among all candidates, our trackers considered the appearance similarity of each detection to the target appearance model and the distance of the detection to the center of the image. Our appearance model relied on histograms computed in the HSV color space.

In Sec. 3.2.1, the appearance model of the tracker is implemented as a dynamic memory that stores color histograms of past detections. The memory is managed in such a way that only the most discriminative color histograms are retained in memory whenever a new appearance representation is available. Moreover, to decide if updating the memory or not, a threshold on the Bhattacharyya distance is computed online by assuming that distances among detections of the same target follows a Gaussian probability distribution.

In Sec. 3.2.2, another tracking-by-detection algorithm for 360° videos was presented. This tracker used the pre-trained deep ConvNet, Mask R-CNN, to strengthen the appearance model of the target by means of the binary segmentation mask of the objects detected by the network. We obtained a very reliable target color distribution, since the cluttered background problem was mitigated by the use of the predicted segmentation masks. This tracker showed that the application of a ConvNet allows to obtain better results. The tracker is not able to work in real-time due to ConvNet's computational cost but further studies can be done to try to simplify the network for real-time uses. An improvement to take into account is to avoid the use of color histogram to represent the appearance of the target and to derive an appearance model from the features extracted by the network itself. Besides the learning of target appearance or similarity metrics by means of CNNs, these networks could also be used to model the motion dynamics of a target. Motion dynamics could be modeled by a deep CNN trained to regress the adjustments of the pan, tilt and zoom parameters in order to maintain the target at the center of the vPTZ camera view. The vPTZ framework adopted the naive but accurate solution of mapping the spherical surface of the equirectangular image onto a tangent plane, to generate different projections. An alternative to this approach could also be to learn a CNN that processes a 360° image in its equirectangular projection but mimics the *flat* filter responses that an existing network would produce on all tangent plane projections for the original spherical image.

In Sec. 3.3 a framework to track pedestrians in 360° videos by using particle filter and generating virtual PTZ cameras was proposed. Rather than formulating the tracking problem as the one of controlling a single vPTZ camera, as in our previous tracking algorithms presented in Secs.3.2.1 and 3.2.2, we assumed that the vPTZ camera to foveate on the target at a time instant is a random variable, and we approximated the posterior distribution of the underlying stochastic process by means of a discrete set of vPTZ camera views. The visual tracking problem was formulated as the one of selecting, at each time, the vPTZ camera to foveate on the target from an unlimited set of simultaneously generated vPTZ camera views. To model the posterior distribution over vPTZ camera views, we adopted the particle filter framework in which each particle is a vPTZ camera with specific pan, tilt and zoom parameters. Such cameras are weighted and used, frame-by-frame, to estimate the vPTZ camera to track the target. In contrast to other particle filter based visual tracking approaches, in which the particle state includes the target's location on the image, in this method the particle state includes the parameters of a vPTZ camera. In our method [16], The particle filter framework improved the estimation of the target location in complex environments and allowed us to track in non-static scenes, in which non-linearity and multi-modality are likely to be significant. The downside effect of this method is the increasing computational burden due to the number of particles.

In future work, we will study the possibility to use offline trained deep learning models to establish matches between the target and the candidate targets. We aim at improving our tracking algorithm in order to include motion prediction in the framework, and to extend the approach to multi-target tracking so to decrease the number of ID switches. The use of vPTZ camera allows to implement the multi-target tracking with no effort. In fact, a vPTZ tracker could be initialized for each target and information about the estimated targets' locations on the sphere could be shared among trackers in order to disambiguate among them. We will also focus on improving the observation likelihood model and, in general, the appearance model used to describe the target in order to make it adaptive. Furthermore, we will consider applications related to 360° camera network.

6.2 Human Behavior Characterization

In Sec. 4.1, a method for representing actions in terms of multinomial distributions of frequent sequential patterns of different length was presented. The method learns a codebook of frequent sequential patterns, that are computed by a modified apriori algorithm. We answered to the question how frequent our frequent patterns have to be for being included in the codebook. During the pattern discovery process, all frequent patterns that do not contribute to the generation of longer patterns were added to our codebook. Our experiments showed that the method benefits from ignoring infrequent patterns both in terms of recall and computational complexity, since a more compact sequence descriptor can be obtained with a smaller codebook. However, considering only the most frequent patterns may result in a lost of details of the action representation and, hence, might have a negative impact on the performance. We presented results by validating the method on skeletal data. On the MSRC-12 dataset this method achieved state-of-the-art accuracy values. In future work, we will extensively study the effect of varying some parameters, such as ϵ and ψ, on the performance of the method. The main limitation of this method is that it might not be able to cope with varying execution velocity of the action, which also depends on the subject. Therefore, we also plan to extend our formulation by accounting for the misalignments between patterns and matched temporal window in order to improve the learning of sequential patterns.

In Sec. 4.2 a system aimed at the detection and recognition of change-points in behavioral data related to humans' movements was presented. In particular, the proposed methodology has been used to implement a semi-automated annotation system able of achieving a fine, local characterization of behavioral observations. The system has been tested using a database of behavioral data collected during an imitation experiment involving interactions between children and a virtual avatar acting as a tightrope walker. Such behavioral data have been exploited to evaluate the general performances of the system and its reliability. The performed experiments using data from children with typical development (TD) showed the effectiveness of the system on detecting and identifying change-points against the background and among three classes. Behavioral data from children with developmental coordination disorder (DCD) and autism spectrum disorder (ASD) have been exploited to verify the reliability of the system facing abnormal movements. Notably, despite an expected decrease in performances, the system is still able to obtain an acceptable recognition rate in accord to the particular neurodevelopmental disorders (NDD) analysed. Finally, through an analysis of the impact of the training set size on the performances of the system, we found that it is possible to achieve an F1-score of 90% by supplying a 30% of the number of subjects in the sample set.

The proposed system has not the ambition of operating as a fully automated change-point annotation system, but as a convenient, semi-automated tool for psychologists, psychiatrists, computer scientists, cognitive scientists and other practitioners working on behavior understanding, affective computing, social robotics or, more in general, human-machine interaction, that need an efficient tool for annotating behavioral observations. After the manual annotation of a small, randomized set of behavioral data, the presented system can be employed to complete the annotation of the whole dataset. However, according to the degree of reliability requested by the behavioral analysis, and due to the not negligible presence of errors, especially in case of behavioral anomalies or NDD, the detected change-points should always be reviewed by expert annotators. Despite these limits, the use of the system would result in a fast and efficient workflow for the annotation of behavioral observations.

Consequently, future work will focus on testing the system on other modalities, such as gazing or eye direction, able to describe the human engagement's evolution. At the same time, this tool will be extended to support different modalities to depict variations on emotional states. Such extension will also account for the different sampling rates of the multi-modal sensors used to capture human behaviors, such as skeleton data from RGBD cameras and gaze information from eye trackers (usually faster than RGBD). We will also further investigate domain-adaptation techniques to transfer the system ability of classifying TD data onto NDD data. Finally, we have the ambition of enlarging the testing domains of the presented framework beyond the analysis of human activities by considering, for instance, the analysis of physical, astronomical or financial observations.

Bibliography

[1] Bobick AF, Davis JW. The recognition of human movement using temporal templates. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2001;p. 257–267.

[2] Ahmed MH, Sabir AT. Human Gender Classification Based on Gait Features Using Kinect Sensor. In: 2017 3rd IEEE International Conference on Cybernetics (CYBCONF). IEEE; 2017. p. 1–5.

[3] Chen G, St-Charles PL, Bouachir W, Bilodeau GA, Bergevin R. Reproducible evaluation of pan-tilt-zoom tracking. In: 2015 IEEE International Conference on Image Processing (ICIP). IEEE; 2015. p. 2055–2059.

[4] Xavier J, Gauthier S, Cohen D, Zahoui M, Chetouani M, Villa F, et al. Interpersonal synchronization, motor coordination, and control are impaired during a dynamic imitation task in children with autism spectrum disorder. Frontiers in psychology. 2018;9.

[5] Lo Presti L, La Cascia M. Real-time object detection in embedded video surveillance systems. In: 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services. IEEE; 2008. p. 151–154.

[6] Yilmaz A, Javed O, Shah M. Object tracking: A survey. Acm computing surveys (CSUR). 2006;38(4):13.

[7] Smeulders AW, Chu DM, Cucchiara R, Calderara S, Dehghan A, Shah M. Visual tracking: An experimental survey. IEEE transactions on pattern analysis and machine intelligence. 2013;36(7):1442–1468.

[8] Hu W, Tan T, Wang L, Maybank S. A survey on visual surveillance of object motion and behaviors. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2004;34(3):334–352.

[9] Balaji S, Karthikeyan S. A survey on moving object tracking using image processing. In: 2017 11th international conference on intelligent systems and control (ISCO). IEEE; 2017. p. 469–474.

[10] Zhang HB, Zhang YX, Zhong B, Lei Q, Yang L, Du JX, et al. A comprehensive survey of vision-based human action recognition methods. Sensors. 2019;19(5):1005.

[11] Lo Presti L, La Cascia M. 3D skeleton-based human action classification: A survey. Pattern Recognition. 2016;53:130–147.

[12] Wang H, Schmid C. Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision; 2013. p. 3551–3558.

[13] Laxman S, Sastry PS. A survey of temporal data mining. Sadhana. 2006;31(2):173–198.

[14] Aminikhanghahi S, Cook DJ. A survey of methods for time series change point detection. Knowledge and information systems. 2017;51(2):339–367.

[15] Monteleone V, Lo Presti L, La Cascia M. Pedestrian Tracking in 360 Video by Virtual PTZ Cameras. In: 2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI). IEEE; 2018. p. 1–6.

[16] Monteleone V, Lo Presti L, La Cascia M. Particle filtering for tracking in 360 degrees videos using virtual PTZ cameras. In: International Conference on Image Analysis and Processing. Springer; 2019. p. 71–81.

[17] Monteleone V, Lo Presti L, La Cascia M. HoP: Histogram of Patterns for Human Action Representation. In: International Conference on Image Analysis and Processing. Springer; 2017. p. 457–468.

[18] Vinciarelli HPPDS Pantic. Bridging the gap between social animal and unsocial machine: A survey of social signal processing. IEEE Transactions on Affective Computing. 2011;3(1):69–87.

[19] Ghidoni S, Anzalone SM, Munaro M, Michieletto S, Menegatti E. A distributed perception infrastructure for robot assisted living. Robotics and Autonomous Systems. 2014;62(9):1316–1328.

[20] Fei-Fei L, Perona P. A bayesian hierarchical model for learning natural scene categories. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 2. IEEE; 2005. p. 524–531.

[21] Schuldt C, Laptev I, Caputo B. Recognizing human actions: a local SVM approach. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.. vol. 3. IEEE; 2004. p. 32–36.

[22] Niebles JC, Wang H, Fei-Fei L. Unsupervised learning of human action categories using spatial-temporal words. International journal of computer vision. 2008;79(3):299–318.

[23] Karaman S, Seidenari L, Bagdanov AD, Del Bimbo A. L1-regularized logistic regression stacking and transductive crf smoothing for action recognition in video. In: ICCV workshop on action recognition with a large number of classes. vol. 13; 2013. p. 14.

[24] Peng X, Wang L, Wang X, Qiao Y. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. Computer Vision and Image Understanding. 2016;150:109–125.

[25] Murthy O, Goecke R. Ordered trajectories for large scale human action recognition. In: Proceedings of the IEEE international conference on computer vision workshops; 2013. p. 412–419.

[26] Wang H, Kläser A, Schmid C, Liu CL. Dense trajectories and motion boundary descriptors for action recognition. International journal of computer vision. 2013;103(1):60–79.

[27] Peng X, Qiao Y, Peng Q, Qi X. Exploring Motion Boundary based Sampling and Spatial-Temporal Context Descriptors for Action Recognition. In: BMVC. vol. 20; 2013. p. 93–96.

[28] Laptev I, Marszałek M, Schmid C, Rozenfeld B. Learning realistic human actions from movies; 2008. .

[29] Valera M, Velastin SA. Intelligent distributed surveillance systems: a review. IEE Proceedings-Vision, Image and Signal Processing. 2005;152(2):192–204.

[30] Liu H, Chen S, Kubota N. Intelligent video systems and analytics: A survey. IEEE Transactions on Industrial Informatics. 2013;9(3):1222–1233.

[31] Sedky MH, Moniri M, Chibelushi CC. Classification of smart video surveillance systems for commercial applications. In: IEEE Conference on Advanced Video and Signal Based Surveillance, 2005. IEEE; 2005. p. 638–643.

[32] Hubel DH, Wiesel TN. Receptive fields and functional architecture of monkey striate cortex. The Journal of physiology. 1968;195(1):215–243.

[33] Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics. 1980;36(4):193–202.

[34] Tsotsos JK, Culhane SM, Wai WYK, Lai Y, Davis N, Nuflo F. Modeling visual attention via selective tuning. Artificial intelligence. 1995;78(1-2):507–545.

[35] Olshausen BA, Field DJ. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature. 1996;381(6583):607.

[36] Riesenhuber M, Poggio T. Are cortical models really bound by the "binding problem"? Neuron. 1999;24(1):87–93.

[37] Poppe R. Vision-based human motion analysis: An overview. Computer vision and image understanding. 2007;108(1-2):4–18.

[38] Gibson JJ. The perception of the visual world. 1950;.

[39] Horn BK, Schunck BG. Determining optical flow. Artificial intelligence. 1981;17(1-3):185–203.

[40] Verri A, Poggio T. Motion field and optical flow: Qualitative properties. IEEE Transactions on pattern analysis and machine intelligence. 1989;11(5):490–498.

[41] Gouda R, Nayak JS. Survey on Pedestrian Detection, Classification and Tracking;.

[42] Parekh HS, Thakore DG, Jaliya UK. A survey on object detection and tracking methods. International Journal of Innovative Research in Computer and Communication Engineering. 2014;2(2):2970–2979.

[43] Ojha S, Sakhare S. Image processing techniques for object tracking in video surveillance-A survey. In: 2015 International Conference on Pervasive Computing (ICPC). IEEE; 2015. p. 1–6.

[44] Zou Z, Shi Z, Guo Y, Ye J. Object Detection in 20 Years: A Survey. arXiv preprint arXiv:190505055. 2019;.

[45] Huang P, Meng Z, Guo J, Zhang F. Tethered Space Robot: Dynamics, Measurement, and Control. Academic Press; 2017.

[46] Pflugfelder RP. A comparison of visual feature tracking methods for traffic monitoring. ÖGAI Journal. 2000;19(4):15–22.

[47] Veenman CJ, Reinders MJ, Backer E. Resolving motion correspondence for densely moving points. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2001;23(1):54–72.

[48] Serby D, Meier E, Van Gool L. Probabilistic object tracking using multiple features. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.. vol. 2. IEEE; 2004. p. 184–187.

[49] Comaniciu D, Ramesh V, Meer P. Kernel-based object tracking. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2003;p. 564–575.

[50] Chang F, Chen CJ, Lu CJ. A linear-time component-labeling algorithm using contour tracing technique. computer vision and image understanding. 2004;93(2):206–220.

[51] Xu N, Ahuja N. Object contour tracking using graph cuts based active contours. In: Proceedings. International Conference on Image Processing. vol. 3. IEEE; 2002. p. III–III.

[52] Deori B, Thounaojam DM. A survey on moving object tracking in video. International Journal on Information Theory (IJIT). 2014;3(3):31–46.

[53] Wang X, Han TX, Yan S. An HOG-LBP human detector with partial occlusion handling. In: 2009 IEEE 12th international conference on computer vision. IEEE; 2009. p. 32–39.

[54] Nummiaro K, Koller-Meier E, Van Gool L. An adaptive color-based particle filter. Image and vision computing. 2003;21(1):99–110.

[55] Dalal N, Triggs B. Histograms of oriented gradients for human detection; 2005. .

[56] Zhu SC, Yuille A. Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. IEEE Transactions on Pattern Analysis & Machine Intelligence. 1996;p. 884–900.

[57] Paragios N, Deriche R. Geodesic active regions and level set methods for supervised texture segmentation. International Journal of Computer Vision. 2002;46(3):223–247.

[58] Elgammal A, Duraiswami R, Harwood D, Davis LS. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE. 2002;90(7):1151–1163.

[59] Fieguth P, Terzopoulos D. Color-based tracking of heads and other mobile objects at video frame rates. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition. IEEE; 1997. p. 21–27.

[60] Edwards GJ, Taylor CJ, Cootes TF. Interpreting face images using active appearance models. In: Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition. IEEE; 1998. p. 300–305.

[61] Moghaddam B, Pentland A. Probabilistic visual learning for object representation. IEEE Transactions on pattern analysis and machine intelligence. 1997;19(7):696–710.

[62] Black MJ, Jepson AD. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. International Journal of Computer Vision. 1998;26(1):63–84.

[63] Avidan S. Support vector tracking. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. vol. 1. IEEE; 2001. p. I–I.

[64] Park S, Aggarwal JK. A hierarchical Bayesian network for event recognition of human actions and interactions. Multimedia systems. 2004;10(2):164–179.

[65] Dosovitskiy A, Tobias Springenberg J, Brox T. Learning to generate chairs with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2015. p. 1538–1546.

[66] Leal-Taixé L, Canton-Ferrer C, Schindler K. Learning by tracking: Siamese CNN for robust target association. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops; 2016. p. 33–40.

[67] Held D, Thrun S, Savarese S. Learning to track at 100 fps with deep regression networks. In: European Conference on Computer Vision. Springer; 2016. p. 749–765.

[68] Isard M, Blake A. Condensation—conditional density propagation for visual tracking. International journal of computer vision. 1998;29(1):5–28.

[69] Lucas BD, Kanade T, et al. An iterative image registration technique with an application to stereo vision. 1981;.

[70] Tomasi C, Detection TK. Tracking of point features. Tech. Rep. CMU-CS-91-132, Carnegie Mellon University; 1991.

[71] Comaniciu D, Ramesh V, Meer P. Real-time tracking of non-rigid objects using mean shift. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662). vol. 2. IEEE; 2000. p. 142–149.

[72] Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2002;p. 603–619.

[73] Kalman RE. A new approach to linear filtering and prediction problems. Journal of basic Engineering. 1960;82(1):35–45.

[74] Breitenstein MD, Reichlin F, Leibe B, Koller-Meier E, Van Gool L. Robust tracking-by-detection using a detector confidence particle filter. In: 2009 IEEE 12th International Conference on Computer Vision. IEEE; 2009. p. 1515–1522.

[75] Yu Q, Medioni G, Cohen I. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2007. p. 1–8.

[76] Yang B, Nevatia R. Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2012. p. 1918–1925.

[77] Ljung L. Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. IEEE Transactions on Automatic Control. 1979;24(1):36–50.

[78] Evensen G. The ensemble Kalman filter: Theoretical formulation and practical implementation. Ocean dynamics. 2003;53(4):343–367.

[79] Wan EA, Van Der Merwe R. The unscented Kalman filter for nonlinear estimation. In: Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373). Ieee; 2000. p. 153–158.

[80] Khan Z, Balch T, Dellaert F. An MCMC-based particle filter for tracking multiple interacting targets. In: European Conference on Computer Vision. Springer; 2004. p. 279–290.

[81] Yang M, Lv F, Xu W, Gong Y, et al. Detection driven adaptive multi-cue integration for multiple human tracking. In: 2009 IEEE 12th International Conference on Computer Vision (ICCV). IEEE; 2009. p. 1554–1561.

[82] Jin Y, Mokhtarian F. Variational particle filter for multi-object tracking. In: 2007 IEEE 11th International Conference on Computer Vision. IEEE; 2007. p. 1–8.

[83] Doucet A, Godsill S, Andrieu C. On sequential Monte Carlo sampling methods for Bayesian filtering. Statistics and computing. 2000;10(3):197–208.

[84] Gordon NJ, Salmond DJ, Smith AF. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: IEE proceedings F (radar and signal processing). vol. 140. IET; 1993. p. 107–113.

[85] Isard M, Blake A. Contour tracking by stochastic propagation of conditional density. In: European conference on computer vision. Springer; 1996. p. 343–356.

[86] MacCormick J, Blake A. A probabilistic exclusion principle for tracking multiple objects. International Journal of Computer Vision. 2000;39(1):57–71.

[87] Beymer D, McLauchlan P, Coifman B, Malik J. A real-time computer vision system for measuring traffic parameters. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition. IEEE; 1997. p. 495–501.

[88] Greiffenhagen M, Ramesh V, Comaniciu D, Niemann H. Statistical modeling and performance characterization of a real-time dual camera surveillance system.

In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662). vol. 2. IEEE; 2000. p. 335–342.

[89] Milan A, Rezatofighi SH, Dick A, Reid I, Schindler K. Online multi-target tracking using recurrent neural networks. In: Thirty-First AAAI Conference on Artificial Intelligence; 2017. .

[90] Sadeghian A, Alahi A, Savarese S. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In: Proceedings of the IEEE International Conference on Computer Vision; 2017. p. 300–311.

[91] Fragkiadaki K, Levine S, Felsen P, Malik J. Recurrent network models for human dynamics. In: Proceedings of the IEEE International Conference on Computer Vision; 2015. p. 4346–4354.

[92] Stalder S, Grabner H, Van Gool L. Dynamic objectness for adaptive tracking. In: Asian conference on computer vision. Springer; 2012. p. 43–56.

[93] Luo W, Xing J, Milan A, Zhang X, Liu W, Zhao X, et al. Multiple object tracking: A literature review. arXiv preprint arXiv:14097618. 2014;.

[94] Shi J, Tomasi C. Good features to track. Cornell University; 1993.

[95] Branson S, Wah C, Schroff F, Babenko B, Welinder P, Perona P, et al. Visual recognition with humans in the loop. In: European Conference on Computer Vision. Springer; 2010. p. 438–451.

[96] Bazzani L, Cristani M, Perina A, Murino V. Multiple-shot person re-identification by chromatic and epitomic analyses. Pattern Recognition Letters. 2012;33(7):898–903.

[97] Özuysal M, Lepetit V, Fleuret F, Fua P. Feature harvesting for tracking-by-detection. In: European conference on computer vision. Springer; 2006. p. 592–605.

[98] Leibe B, Schindler K, Cornelis N, Van Gool L. Coupled object detection and tracking from static cameras and moving vehicles. IEEE transactions on pattern analysis and machine intelligence. 2008;30(10):1683–1698.

[99] Hager GD, Belhumeur PN. Efficient region tracking with parametric models of geometry and illumination. IEEE Transactions on Pattern Analysis & Machine Intelligence. 1998;p. 1025–1039.

[100] Baker S, Matthews I. Lucas-kanade 20 years on: A unifying framework. International journal of computer vision. 2004;56(3):221–255.

[101] Matthews L, Ishikawa T, Baker S. The template update problem. IEEE transactions on pattern analysis and machine intelligence. 2004;26(6):810–815.

[102] Koller D, Weber J, Malik J. Robust multiple car tracking with occlusion reasoning. In: European Conference on Computer Vision. Springer; 1994. p. 189–196.

[103] Betke M, Haritaoglu E, Davis LS. Real-time multiple vehicle detection and tracking from a moving vehicle. Machine vision and applications. 2000;12(2):69–83.

[104] Tang S, Andriluka M, Andres B, Schiele B. Multiple people tracking by lifted multicut and person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 3539–3548.

[105] Oh S, Russell S, Sastry S. Markov chain Monte Carlo data association for multi-target tracking. IEEE Transactions on Automatic Control. 2009;54(3):481–497.

[106] Yu Q, Medioni G. Multiple-target tracking by spatiotemporal monte carlo markov chain data association. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2008;31(12):2196–2210.

[107] Zhang J, Lo Presti L, Sclaroff S. Online multi-person tracking by tracker hierarchy. In: 2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance. IEEE; 2012. p. 379–385.

[108] Lo Presti L, Morana M, La Cascia M. A data association algorithm for people re-identification in photo sequences. In: 2010 IEEE International Symposium on Multimedia. IEEE; 2010. p. 318–323.

[109] Lo Presti L, La Cascia M. An on-line learning method for face association in personal photo collection. Image and Vision Computing. 2012;30(4-5):306–316.

[110] Ross DA, Lim J, Lin RS, Yang MH. Incremental learning for robust visual tracking. International journal of computer vision. 2008;77(1-3):125–141.

[111] Avidan S. Ensemble tracking. IEEE transactions on pattern analysis and machine intelligence. 2007;29(2):261–271.

[112] Breitenstein MD, Reichlin F, Leibe B, Koller-Meier E, Van Gool L. Online multiperson tracking-by-detection from a single, uncalibrated camera. IEEE transactions on pattern analysis and machine intelligence. 2010;33(9):1820–1833.

[113] Andriyenko A, Roth S, Schindler K. An analytical formulation of global occlusion reasoning for multi-target tracking. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). IEEE; 2011. p. 1839–1846.

[114] Viola P, Jones M, et al. Rapid object detection using a boosted cascade of simple features. CVPR (1). 2001;1(511-518):3.

[115] Ren S, He K, Girshick R, Sun J. Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems; 2015. p. 91–99.

[116] Dollár P, Wojek C, Schiele B, Perona P. Pedestrian detection: A benchmark. 2009;.

[117] Wu J, Geyer C, Rehg JM. Real-time human detection using contour cues. In: 2011 IEEE International Conference on Robotics and Automation. IEEE; 2011. p. 860–867.

[118] Swain MJ, Ballard DH. Color indexing. International journal of computer vision. 1991;7(1):11–32.

[119] Chang F, Zhang G, Wang X, Chen Z. PTZ camera target tracking in large complex scenes. In: 2010 8th World Congress on Intelligent Control and Automation. IEEE; 2010. p. 2914–2918.

[120] Kang S, Paik JK, Koschan A, Abidi BR, Abidi MA. Real-time video tracking using PTZ cameras. In: Sixth International Conference on Quality Control by Artificial Vision. vol. 5132. International Society for Optics and Photonics; 2003. p. 103–111.

[121] Girshick R. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision; 2015. p. 1440–1448.

[122] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proc. of the conf. on computer vision and pattern recognition (CVPR); 2014. p. 580–587.

[123] He K, Zhang X, Ren S, Sun J. Spatial pyramid pooling in deep convolutional networks for visual recognition. Trans on pattern analysis and machine intelligence (PAMI). 2015;37(9):1904–1916.

[124] Dai J, Li Y, He K, Sun J. R-fcn: Object detection via region-based fully convolutional networks. In: Advances in neural information processing systems; 2016. p. 379–387.

[125] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: Proc. of the conf. on computer vision and pattern recognition (CVPR); 2016. p. 779–788.

[126] Gidaris S, Komodakis N. Object detection via a multi-region and semantic segmentation-aware cnn model. In: Proc. of the Int. Conf. on Computer Vision (ICCV); 2015. p. 1134–1142.

[127] He K, Gkioxari G, Dollár P, Girshick R. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision; 2017. p. 2961–2969.

[128] Bertinetto L, Valmadre J, Henriques JF, Vedaldi A, Torr PH. Fully-convolutional siamese networks for object tracking. In: European conference on computer vision. Springer; 2016. p. 850–865.

[129] Brand M, Oliver N, Pentland A. Coupled hidden Markov models for complex action recognition. In: cvpr. vol. 97; 1997. p. 994.

[130] Du Y, Chen F, Xu W. Human interaction representation and recognition through motion decomposition. IEEE Signal Processing Letters. 2007;14(12):952–955.

[131] Bodor R, Jackson B, Papanikolopoulos N. Vision-based human tracking and activity recognition. In: Proc. of the 11th Mediterranean Conf. on Control and Automation. vol. 1; 2003. .

[132] Duong TV, Bui HH, Phung DQ, Venkatesh S. Activity recognition and abnormality detection with the switching hidden semi-markov model. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 1. IEEE; 2005. p. 838–845.

[133] Kuo YM, Lee JS, Chung PC. A visual context-awareness-based sleeping-respiration measurement system. IEEE Transactions on Information Technology in Biomedicine. 2009;14(2):255–265.

[134] Ke SR, Thuc HLU, Lee YJ, Hwang JN, Yoo JH, Choi KH. A review on video-based human activity recognition. Computers. 2013;2(2):88–131.

[135] Lara OD, Labrador MA. A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials. 2012;15(3):1192–1209.

[136] Aggarwal JK, Ryoo MS. Human activity analysis: A review. ACM Computing Surveys (CSUR). 2011;43(3):16.

[137] Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, et al. Real-time human pose recognition in parts from single depth images. In: CVPR 2011. Ieee; 2011. p. 1297–1304.

[138] Chang CC, Lin CJ. LIBSVM: A library for support vector machines. ACM transactions on intelligent systems and technology (TIST). 2011;2(3):27.

[139] Boser BE, Guyon IM, Vapnik VN. A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory. ACM; 1992. p. 144–152.

[140] Ye M, Zhang Q, Wang L, Zhu J, Yang R, Gall J. A survey on human motion analysis from depth data. In: Time-of-flight and depth imaging. sensors, algorithms, and applications. Springer; 2013. p. 149–187.

[141] Blank M, Gorelick L, Shechtman E, Irani M, Basri R. Actions as space-time shapes. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 2. IEEE; 2005. p. 1395–1402.

[142] Shechtman E, Irani M. Matching Local Self-Similarities across Images and Videos. In: CVPR. vol. 2. Minneapolis, MN; 2007. p. 3.

[143] Hoey J. Hierarchical unsupervised learning of facial expression categories. In: Proceedings IEEE Workshop on Detection and Recognition of Events in Video. IEEE; 2001. p. 99–106.

[144] Zhu Y, Zhao X, Fu Y, Liu Y. Sparse coding on local spatial-temporal volumes for human action recognition. In: Asian conference on computer vision. Springer; 2010. p. 660–671.

[145] Demirdjian D, Wang S. Recognition of temporal events using multiscale bags of features. In: 2009 IEEE Workshop on Computational Intelligence for Visual Intelligence. IEEE; 2009. p. 8–13.

[146] Zhao X, Li X, Pang C, Zhu X, Sheng QZ. Online human gesture recognition from motion data streams. In: Proceedings of the 21st ACM international conference on Multimedia. ACM; 2013. p. 23–32.

[147] Wang C, Wang Y, Yuille AL. An approach to pose-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2013. p. 915–922.

[148] Wang J, Liu Z, Wu Y, Yuan J. Mining actionlet ensemble for action recognition with depth cameras. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2012. p. 1290–1297.

[149] Gavrilov Z, Sclaroff S, Neidle C, Dickinson SJ. Detecting Reduplication in Videos of American Sign Language. In: LREC; 2012. p. 3767–3773.

[150] Tang K, Fei-Fei L, Koller D. Learning latent temporal structure for complex event detection. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2012. p. 1250–1257.

[151] Sminchisescu C, Kanaujia A, Metaxas D. Conditional models for contextual human motion recognition. Computer Vision and Image Understanding. 2006;104(2-3):210–220.

[152] Thi TH, Zhang J, Cheng L, Wang L, Satoh S. Human action recognition and localization in video using structured learning of local space-time features. In: 2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance. IEEE; 2010. p. 204–211.

[153] Laptev I. On space-time interest points. International journal of computer vision. 2005;64(2-3):107–123.

[154] LeCun Y, Bottou L, Bengio Y, Haffner P, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278–2324.

[155] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. science. 2006;313(5786):504–507.

[156] Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. Neural computation. 2006;18(7):1527–1554.

[157] Bengio Y, et al. Learning deep architectures for AI. Foundations and trends® in Machine Learning. 2009;2(1):1–127.

[158] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, et al. Backpropagation applied to handwritten zip code recognition. Neural computation. 1989;1(4):541–551.

[159] Graves A, Schmidhuber J. Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in neural information processing systems; 2009. p. 545–552.

[160] Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE. A survey of deep neural network architectures and their applications. Neurocomputing. 2017;234:11–26.

[161] Wang J, Chen Y, Hao S, Peng X, Hu L. Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters. 2019;119:3–11.

[162] Zeng M, Nguyen LT, Yu B, Mengshoel OJ, Zhu J, Wu P, et al. Convolutional neural networks for human activity recognition using mobile sensors. In: 6th International Conference on Mobile Computing, Applications and Services. IEEE; 2014. p. 197–205.

[163] Simonyan K, Zisserman A. Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems; 2014. p. 568–576.

[164] Ji S, Xu W, Yang M, Yu K. 3D convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence. 2012;35(1):221–231.

[165] Tran D, Bourdev L, Fergus R, Torresani L, Paluri M. Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision; 2015. p. 4489–4497.

[166] Du Y, Wang W, Wang L. Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 1110–1118.

[167] Park E, Han X, Berg TL, Berg AC. Combining multiple sources of knowledge in deep cnns for action recognition. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE; 2016. p. 1–8.

[168] Rahmani H, Mian A. 3D action recognition from novel viewpoints. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 1506–1515.

[169] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation. 1997;9(8):1735–1780.

[170] Tallec C, Ollivier Y. Can recurrent neural networks warp time? arXiv preprint arXiv:180411188. 2018;.

[171] Zhu W, Lan C, Xing J, Zeng W, Li Y, Shen L, et al. Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In: Thirtieth AAAI Conference on Artificial Intelligence; 2016. .

[172] Lefebvre G, Berlemont S, Mamalet F, Garcia C. BLSTM-RNN based 3D gesture classification. In: International conference on artificial neural networks. Springer; 2013. p. 381–388.

[173] Liu J, Shahroudy A, Xu D, Wang G. Spatio-temporal lstm with trust gates for 3d human action recognition. In: European Conference on Computer Vision. Springer; 2016. p. 816–833.

[174] Li X, Zhang Y, Zhang J. Improved key poses model for skeleton-based action recognition. In: Pacific Rim Conference on Multimedia. Springer; 2017. p. 358–367.

[175] Liu Z, Zhu J, Bu J, Chen C. A survey of human pose estimation: the body parts parsing based methods. Journal of Visual Communication and Image Representation. 2015;32:10–19.

[176] Chen L, Wei H, Ferryman J. A survey of human motion analysis using depth imagery. Pattern Recognition Letters. 2013;34(15):1995–2006.

[177] Yao A, Gall J, Fanelli G, Van Gool L. Does human action recognition benefit from pose estimation? In: BMVC 2011-Proceedings of the British Machine Vision Conference 2011; 2011. .

[178] Müller M, Röder T, Clausen M. Efficient content-based retrieval of motion capture data. In: ACM Transactions on Graphics (ToG). vol. 24. ACM; 2005. p. 677–685.

[179] Ramanan D, Forsyth DA, Zisserman A. Strike a pose: Tracking people by finding stylized poses. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 1. IEEE; 2005. p. 271–278.

[180] Cao Z, Hidalgo G, Simon T, Wei SE, Sheikh Y. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. arXiv preprint arXiv:181208008. 2018;.

[181] Pishchulin L, Insafutdinov E, Tang S, Andres B, Andriluka M, Gehler PV, et al. Deepcut: Joint subset partition and labeling for multi person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 4929–4937.

[182] Levitis DA, Lidicker Jr WZ, Freund G. Behavioural biologists do not agree on what constitutes behaviour. Animal behaviour. 2009;78(1):103–110.

[183] Thagard P. Mind: Introduction to cognitive science. vol. 17. MIT press Cambridge, MA; 2005.

[184] Corr PJ, Matthews G. The Cambridge handbook of personality psychology. Cambridge University Press Cambridge; 2009.

[185] Association AP, et al. Diagnostic and statistical manual of mental disorders. BMC Med. 2013;17:133–137.

[186] Breazeal CL. Designing sociable robots. MIT press; 2004.

[187] Sun R. The Cambridge handbook of computational psychology. Cambridge University Press; 2008.

[188] Anzalone SM, Boucenna S, Ivaldi S, Chetouani M. Evaluating the engagement with social robots. International Journal of Social Robotics. 2015;7(4):465–478.

[189] Vinciarelli A, Pantic M, Bourlard H. Social signal processing: Survey of an emerging domain. Image and vision computing. 2009;27(12):1743–1759.

[190] Ogden L. Collaborative tasks, collaborative children: An analysis of reciprocity during peer interaction at Key Stage 1. British Educational Research Journal. 2000;26(2):211–226.

[191] Burgos-Artizzu XP, Dollár P, Lin D, Anderson DJ, Perona P. Social behavior recognition in continuous video. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2012. p. 1322–1329.

[192] Rehg RRCSEOLK Abowd, et al. Decoding children's social behavior. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2013. p. 3414–3421.

[193] Mahdhaoui CSGPLAMMC Chetouani. Computerized home video detection for motherese may help to study impaired interaction between infants who become autistic and their parents. International Journal of Methods in Psychiatric Research. 2011;20(1):e6–e18.

[194] Funes Mora KA, Nguyen L, Gatica-Perez D, Odobez JM. A semi-automated system for accurate gaze coding in natural dyadic interactions. In: Proceedings of the 15th ACM on International conference on multimodal interaction. ACM; 2013. p. 87–90.

[195] Fung M, Jin Y, Zhao R, Hoque ME. ROC speak: semi-automated personalized feedback on nonverbal behavior from recorded videos. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing. ACM; 2015. p. 1167–1178.

[196] Salam H, Celiktutan O, Hupont I, Gunes H, Chetouani M. Fully automatic analysis of engagement and its relationship to personality in human-robot interactions. IEEE Access. 2016;5:705–721.

[197] Shou Z, Pan J, Chan J, Miyazawa K, Mansour H, Vetro A, et al. Online detection of action start in untrimmed, streaming videos. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. p. 534–551.

[198] Wang B, Hoai M. Back to the beginning: Starting point detection for early recognition of ongoing human actions. Computer Vision and Image Understanding. 2018;175:24–31.

[199] Lo Presti L, La Cascia M, Sclaroff S, Camps O. Hankelet-based dynamical systems modeling for 3D action recognition. Image and Vision Computing. 2015;44:29–43.

[200] Boettcher M. Contrast and change mining. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2011;1(3):215–230.

[201] Hido S, Idé T, Kashima H, Kubo H, Matsuzawa H. Unsupervised change analysis using supervised learning. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer; 2008. p. 148–159.

[202] Kifer D, Ben-David S, Gehrke J. Detecting change in data streams. In: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30. VLDB Endowment; 2004. p. 180–191.

[203] Iniesta R, Stahl D, McGuffin P. Machine learning, statistical learning and the future of biological research in psychiatry. Psychological medicine. 2016;46(12):2455–2465.

[204] Lo Presti L, Sclaroff S, Rozga A. Joint alignment and modeling of correlated behavior streams. In: Proceedings of the IEEE International Conference on Computer Vision Workshops; 2013. p. 730–737.

[205] Achard C, Qu X, Mokhber A, Milgram M. A novel approach for recognition of human actions with semi-global features. Machine Vision and Applications. 2008;19(1):27–34.

[206] Senger L, Schröer M, Metzen JH, Kirchner EA. Velocity-based multiple change-point inference for unsupervised segmentation of human movement behavior. In: 2014 22nd International Conference on Pattern Recognition. IEEE; 2014. p. 4564–4569.

[207] Ambady N, Rosenthal R. Thin slices of expressive behavior as predictors of interpersonal consequences: A meta-analysis. Psychological bulletin. 1992;111(2):256.

[208] Pentland A. A computational model of social signalin. In: 18th International Conference on Pattern Recognition (ICPR'06). vol. 1. IEEE; 2006. p. 1080–1083.

[209] Liu T, Kappas A. Predicting Engagement Breakdown in HRI Using Thin-slices of Facial Expressions. In: Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence; 2018. .

[210] Anzalone SM, Varni G, Ivaldi S, Chetouani M. Automated prediction of extraversion during human–humanoid interaction. International Journal of Social Robotics. 2017;9(3):385–399.

[211] Kanda T, Sato R, Saiwaki N, Ishiguro H. A two-month field trial in an elementary school for long-term human–robot interaction. IEEE Transactions on robotics. 2007;23(5):962–971.

[212] Ivaldi S, Anzalone SM, Rousseau W, Sigaud O, Chetouani M. Robot initiative in a team learning task increases the rhythm of interaction but not the perceived engagement. Frontiers in neurorobotics. 2014;8:5.

[213] Ball K, Webster F. The intensification of surveillance. 2003;.

[214] Aghajan H, Cavallaro A. Multi-camera networks: principles and applications. Academic press; 2009.

[215] Qureshi FZ, Terzopoulos D. Proactive PTZ camera control. In: Distributed Video Sensor Networks. Springer; 2011. p. 273–287.

[216] Salvagnini P, Cristani M, Del Bue A, Murino V. An experimental framework for evaluating PTZ tracking algorithms. In: International Conference on Computer Vision Systems. Springer; 2011. p. 81–90.

[217] Greco L, La Cascia M. 360 Tracking Using a Virtual PTZ Camera. In: International Conference on Image Analysis and Processing. Springer; 2017. p. 62–72.

[218] Liu Z, Cohen M. Head-size equalization for better visual perception of video conferencing. In: 2005 IEEE International Conference on Multimedia and Expo. IEEE; 2005. p. 4–pp.

[219] Kumar CB, Potnis A, Gupta S. Video conferencing system for distance education. In: 2015 IEEE UP Section Conference on Electrical Computer and Electronics (UPCON). IEEE; 2015. p. 1–6.

[220] Gatica-Perez D, Lathoud G, Odobez JM, McCowan I. Audiovisual probabilistic tracking of multiple speakers in meetings. IEEE Transactions on Audio, Speech, and Language Processing. 2007;15(2):601–616.

[221] Kwiatek K, Woolner M. Transporting the viewer into a 360 heritage story: Panoramic interactive narrative presented on a wrap-around screen. In: 2010 16th International Conference on Virtual Systems and Multimedia. IEEE; 2010. p. 234–241.

[222] Sandnes FE. Communicating panoramic 360 degree immersed experiences: a simple technique for sketching in 3D. In: International Conference on Universal Access in Human-Computer Interaction. Springer; 2016. p. 338–346.

[223] Izard SG, Méndez JAJ, García-Peñalvo FJ, López MJ, Vázquez FP, Ruisoto P. 360 vision applications for medical training. In: Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality. ACM; 2017. p. 55.

[224] Snyder JP. Map projections–A working manual. vol. 1395. US Government Printing Office; 1987.

[225] mathworld. Gnomonic Projection; 2019. Available from: `http://mathworld.wolfram.com/GnomonicProjection.html`.

[226] Su YC, Grauman K. Learning spherical convolution for fast features from 360 imagery. In: Advances in Neural Information Processing Systems; 2017. p. 529–539.

[227] Zhao Q, Zhu C, Dai F, Ma Y, Jin G, Zhang Y. Distortion-aware CNNs for Spherical Images. In: IJCAI; 2018. p. 1198–1204.

[228] Tang Y, Bilodeau GA. Evaluation of trackers for Pan-Tilt-Zoom Scenarios. arXiv preprint arXiv:171104260. 2017;.

[229] Cai C, Liang X, Wang B, Cui Y, Yan Y. A Target Tracking Method Based on KCF for Omnidirectional Vision. In: 2018 37th Chinese Control Conference (CCC). IEEE; 2018. p. 2674–2679.

[230] Delforouzi A, Tabatabaei SAH, Shirahama K, Grzegorzek M. Unknown object tracking in 360-degree camera images. In: 2016 23rd International Conference on Pattern Recognition (ICPR). IEEE; 2016. p. 1798–1803.

[231] Dollar P, Wojek C, Schiele B, Perona P. Pedestrian detection: A benchmark. Computer Vision and Pattern Recognition, 2009. CVPR 2009. In: IEEE Conference on; 2009. p. 304–311.

[232] Mikami D, Otsuka K, Yamato J. Memory-based particle filter for tracking objects with large variation in pose and appearance. In: European Conference on Computer Vision. Springer; 2010. p. 215–228.

[233] Geng Y, Hu HM, Zeng G, Zheng J. A person re-identification algorithm by exploiting region-based feature salience. Journal of Visual Communication and Image Representation. 2015;29:89–102.

[234] Bhattacharyya A. On a measure of divergence between two multinomial populations. Sankhyā: the indian journal of statistics. 1946;p. 401–406.

[235] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems; 2012. p. 1097–1105.

[236] Abdulla W. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. GitHub repository. 2017;.

[237] Ranasinghe S, Al Machot F, Mayr HC. A review on applications of activity recognition systems with regard to performance and evaluation. International Journal of Distributed Sensor Networks. 2016;12(8):1550147716665520.

[238] Leont'ev AN. Activity, consciousness, and personality. 1978;.

[239] Mayr HC, Al Machot F, Michael J, Morak G, Ranasinghe S, Shekhovtsov V, et al. HCM-L: domain-specific modeling for active and assisted living. In: Domain-Specific Conceptual Modeling. Springer; 2016. p. 527–552.

[240] Choi W, Savarese S. A unified framework for multi-target tracking and collective activity recognition. In: European Conference on Computer Vision. Springer; 2012. p. 215–230.

[241] Agrawal R, Imieliński T, Swami A. Mining association rules between sets of items in large databases. In: Acm sigmod record. vol. 22. ACM; 1993. p. 207–216.

[242] Fothergill S, Mentis H, Kohli P, Nowozin S. Instructing people for training gestural interactive systems. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM; 2012. p. 1737–1746.

[243] Agrawal R, Srikant R, et al. Mining sequential patterns. In: icde. vol. 95; 1995. p. 3–14.

[244] Gauthier S, Anzalone SM, Cohen D, Zaoui M, Chetouani M, Villa F, et al. Behavioral Own-Body-Transformations in Children and Adolescents With Typical Development, Autism Spectrum Disorder, and Developmental Coordination Disorder. Frontiers in psychology. 2018;9:676.

[245] Thirioux B, Jorland G, Bret M, Tramus MH, Berthoz A. Walking on a line: A motor paradigm using rotation and reflection symmetry to study mental body transformations. Brain and cognition. 2009;70(2):191–200.

[246] Ingersoll B. The social role of imitation in autism: Implications for the treatment of imitation deficits. Infants & Young Children. 2008;21(2):107–119.

[247] Roeyers H, Van Oost P, Bothuyne S. Immediate imitation and joint attention in young children with autism. Development and psychopathology. 1998;10(3):441–450.

[248] Whiten A, Brown J. Imitation and the reading of other minds: Perspectives from the study of autism, normal children and non-human primates. Intersubjective communication and emotion in early ontogeny. 1998;p. 260–280.

[249] Mazeau M. Troubles neurovisuels et praxiques: un élément déterminant du pronostic à long terme. Médecine thérapeutique/Pédiatrie. 2000;3(4):273–80.

[250] Idé T, Kashima H. Eigenspace-based anomaly detection in computer systems. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2004. p. 440–449.

[251] Kawahara Y, Yairi T, Machida K. Change-point detection in time-series data based on subspace identification. In: Seventh IEEE International Conference on Data Mining (ICDM 2007). IEEE; 2007. p. 559–564.

[252] Ahmed E, Clark A, Mohay G. A novel sliding window based change detection algorithm for asymmetric traffic. In: 2008 IFIP International Conference on Network and Parallel Computing. IEEE; 2008. p. 168–175.

[253] Basseville M, Nikiforov IV, et al. Detection of abrupt changes: theory and application. vol. 104. Prentice Hall Englewood Cliffs; 1993.

[254] James B, James KL, Siegmund D. Tests for a change-point. Biometrika. 1987;74(1):71–83.

[255] Chen J, Gupta AK. On change point detection and estimation. Communications in statistics-simulation and computation. 2001;30(3):665–697.

[256] Willsky A, Jones H. A generalized likelihood ratio approach to the detection and estimation of jumps in linear systems. IEEE Transactions on Automatic control. 1976;21(1):108–112.

[257] Gustafsson F. The marginalized likelihood ratio test for detecting abrupt changes. IEEE Transactions on automatic control. 1996;41(1):66–78.

[258] Young IT. Proof without prejudice: use of the Kolmogorov-Smirnov test for the analysis of histograms from flow systems and other sources. Journal of Histochemistry & Cytochemistry. 1977;25(7):935–941.

[259] Brodsky E, Darkhovsky BS. Nonparametric methods in change point problems. vol. 243. Springer Science & Business Media; 2013.

[260] Carlin BP, Gelfand AE, Smith AF. Hierarchical Bayesian analysis of changepoint problems. Journal of the Royal Statistical Society: Series C (Applied Statistics). 1992;41(2):389–405.

[261] Reddy S, Mun M, Burke J, Estrin D, Hansen M, Srivastava M. Using mobile phones to determine transportation modes. ACM Transactions on Sensor Networks (TOSN). 2010;6(2):13.

[262] Zheng Y, Liu L, Wang L, Xie X. Learning transportation mode from raw gps data for geographic applications on the web. In: Proceedings of the 17th international conference on World Wide Web. ACM; 2008. p. 247–256.

[263] Cleland I, Han M, Nugent C, Lee H, McClean S, Zhang S, et al. Evaluation of prompted annotation of activity data recorded from a smart phone. Sensors. 2014;14(9):15861–15879.

[264] Han M, Lee YK, Lee S, et al. Comprehensive context recognizer based on multimodal sensors in a smartphone. Sensors. 2012;12(9):12588–12605.

[265] Wei L, Keogh E. Semi-supervised time series classification. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2006. p. 748–753.

[266] Zhang K, Zhang L, Yang MH. Real-time compressive tracking. In: European conference on computer vision. Springer; 2012. p. 864–877.

[267] Lacasa L, Luque B, Ballesteros F, Luque J, Nuno JC. From time series to complex networks: The visibility graph. Proceedings of the National Academy of Sciences. 2008;105(13):4972–4975.

[268] Rosenbaum PR. An exact distribution-free test comparing two multivariate distributions based on adjacency. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2005;67(4):515–530.

[269] Mardia KV. Measures of multivariate skewness and kurtosis with applications. Biometrika. 1970;57(3):519–530.

[270] Eager D, Pendrill AM, Reistad N. Beyond velocity and acceleration: jerk, snap and higher derivatives. European Journal of Physics. 2016;37(6):065008.

[271] Hogan N, Sternad D. Sensitivity of smoothness measures to movement duration, amplitude, and arrests. Journal of motor behavior. 2009;41(6):529–534.

[272] Balasubramanian S, Melendez-Calderon A, Burdet E. A robust and sensitive metric for quantifying movement smoothness. IEEE transactions on biomedical engineering. 2011;59(8):2126–2136.

[273] Feuz KD, Cook DJ, Rosasco C, Robertson K, Schmitter-Edgecombe M. Automated detection of activity transitions for prompting. IEEE transactions on human-machine systems. 2014;45(5):575–585.

[274] Desobry F, Davy M, Doncarli C. An online kernel change detection algorithm. IEEE Trans Signal Processing. 2005;53(8-2):2961–2974.

[275] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft coco: Common objects in context. In: European conference on computer vision. Springer; 2014. p. 740–755.

[276] Hussein ME, Torki M, Gowayyed MA, El-Saban M. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In: Twenty-Third International Joint Conference on Artificial Intelligence; 2013. .

[277] Anzalone SM, Tilmont E, Boucenna S, Xavier J, Jouen AL, Bodeau N, et al. How children with autism spectrum disorder behave and explore the 4-dimensional (spatial 3D+ time) environment during a joint attention induction task with a robot. Research in Autism Spectrum Disorders. 2014;8(7):814–826.

[278] Anzalone SM, Xavier J, Boucenna S, Billeci L, Narzisi A, Muratori F, et al. Quantifying patterns of joint attention during human-robot interactions: An application for autism spectrum disorder assessment. Pattern Recognition Letters. 2019;118:42–50.