Universidad Autónoma de Madrid
Escuela politécnica superior

Proyecto fin de carrera

ACCURATE TRAJECTORY REPRESENTATION FOR VIDEO SURVEILLANCE

Ingeniería de Telecomunicación

Gonzalo Varela Bartrina
Enero 2013

ACCURATE TRAJECTORY REPRESENTATION FOR VIDEO SURVEILLANCE

AUTOR: Gonzalo Varela Bartrina
TUTOR: Ebroul Izquierdo
PONENTE: José María Martínez Sánchez

Multimedia and Vision Research Group
School of Electronic Engineering and Computer Science
Queen Mary, University of London
Enero 2013

Resumen

Resumen

El objetivo de este proyecto es el estudio, desarrollo y evaluación de un sistema robusto de representación de trayectorias de objetos para vídeo vigilancia. El sistema debe permitir trabajar de forma flexible con las trayectorias, generando una representación útil para aplicaciones de más alto nivel.
La solución propuesta comprende cuatro fases, cada una con un propósito determinado. La primera fase consiste en un algoritmo que se encarga de realizar las tareas necesarias para extraer las trayectorias de los objetos en movimiento de los vídeos. La segunda fase o comprende una serie de técnicas destinadas a mejorar y depurar la información obtenida en la fase anterior. En la tercera etapa, las trayectorias se dividen para facilitar su posterior uso y análisis. Por último, se desarrolla una representación precisa de las trayectorias que también permite la reconstrucción de las trayectorias originales.
Para la evaluación del sistema se utilizan dos bases de datos distintas, tratando de incluir los escenarios y situaciones más frecuentes en vídeo vigilancia. Cada técnica utilizada en el sistema se evalúa de manera individual, determinando su eficacia para la consecución del objetivo final. Para la correcta evaluación de algunas partes del proyecto ha sido necesario construir una base de datos en la que los objetos de interés han sido anotados manualmente para su comparación con los objetos detectados por el sistema.

Palabras Clave

Video vigilancia, trayectorias de objetos en movimiento, representación de trayectorias, división en subtrayectorias, filtro de Saviztky-Golay, reconstrucción.

Abstract

The main objective of this thesis is to find, develop and evaluate a robust system to represent trajectories extracted from moving objects in surveillance videos. The proposed system should allow a flexible and accurate work with trajectories, generating a useful trajectory representation for high level applications.
The system is divided in four principal steps, each one with its own purpose. The first step is in charge of extract the trajectories of moving objects from the target videos. The second step applies a number of techniques to enhance the data obtained in the previous step. In the third stage, trajectories are split to facilitate further work with trajectories. Finally, in the fourth step, the subpaths previously obtained are characterized by a number of features that allow further trajectory reconstruction.
To evaluate the performance of the proposed system, all the experiments are run over two different datasets. These datasets include the most common scenarios in video surveillance. Each implemented step is analysed individually, determining its effectiveness towards reaching the thesis objectives. A ground truth dataset has been created to evaluate some of the implemented techniques. This ground truth is made of hand annotated objects from several videos of the evaluation datasets.

Key words

Video surveillance, moving objects trajectories, trajectory representation, trajectory division in subpaths, Saviztky-Golay filter, trajectory reconstruction.

Acknowledgements

First of all, I want to thank Ebroul for the opportunity to develop my master thesis abroad and the people of the MMV for their warm welcome. I also want to thank Chema for his support all along the thesis, Paula Fonseca for her aid during my stay in London and Álvaro for his help with the datasets.

A special mention deserves Virginia, without whose help this thesis would never have been finished.

Secondly, I would like to thank all the teachers and professors I've had since school, because they have provided me the tools and knowledge that have brought me here.

Special thanks to my family, which has always supported me, in anything I have decided to undertake. Thanks to my father for inculcate in me curiosity and hunger for knowledge. Thanks to my mother for her example of constant fight and coherence. Thank the both of you, for teach me the value of critical thinking. Finally, say thanks to my grandmother, who has always been an example of courage against all odds.

I want to say, how much I appreciate all my friends. I would like to dedicate a couple of lines to each you, but given that's not possible, I will try my best because you deserve it.
To all the people from the Engineering School of the UAM, for the laughs shared in class or in the labs, but especially for those shared in the café and in the grass. Your friendship and help during these years has been absolutely invaluable.
To all the friends I've made in London, both the spanish and the people from QMR&M Society, thanks for this unforgettable year.
Thanks to the people of the Liceo, for all these years of friendship. Even if we don't see as frequently as I would like, I will always have time for all of you.
To Eva, Miko, Chavo, Aly, Sara and to all my lifelong friends, thank you for all your affection and friendship. Life is much better with you at my side.

Gonzalo Varela Bartrina

January, 2012

Agradecimientos

En primer me gustaría agradecer a Ebroul la posibilidad de desarrollar mi proyecto fuera, lo que me ha proporcionado una experiencia tremendamente enriquecedora. A la gente del MMV por su cálida acogida. Agradecer a Chema sus gestiones desde el principio y su apoyo a lo largo del proyecto, a Paula Fonseca toda su ayuda durante mi estancia en Londres y a Álvaro su ayuda con las bases de datos.

Mención especial merece Virginia, porque sin su ayuda este proyecto jamás habría salido a la luz.

En segundo lugar, quería dar las gracias a los profesores que he tenido a lo largo de mi formación universitaria y escolar. Me habéis proporcionado las herramientas y el conocimiento necesarios. Quiero destacar que esa formación no sólo ha sido científica, sino también humanística y personal. Sin vosotros nunca hubiera llegado hasta aquí.

Quiero mencionar de una forma especial a mi familia, que siempre me ha apoyado en lo que he decidido emprender. A mi padre, por inculcarme la curiosidad, el amor al conocimiento. A mi madre, por su ejemplo de lucha y coherencia. A ambos, por enseñarme el valor del pensamiento crítico. A mi abuela, por ser un pilar de coraje contra viento y marea. Este proyecto es, en parte, vuestro.

A todos mis amigos, ojala pudiera dedicar personalmente unas líneas a cada uno pero, dado el contexto, os habréis de conformar con estas migas.
A la gente de la Escuela, por tantas risas compartidas a la luz de las pantallas de los laboratorios, pero sin duda por aquellas compartidas en la cafetería o el césped. Vuestra compañía y ayuda ha resultado invaluable en estos años de carrera. Gracias a todos.
A todos los amigos que hice en Londres, tanto los españoles, como la gente de la QMR&M Society, gracias por hacer de este año una estancia inolvidable.
A la gente del Liceo por tantos años de amistad y porque sean muchos más. Porque aunque no nos veamos tan a menudo como me gustaría, siempre tendré un hueco para vosotros.
A Eva, Miko, Chavo, Aly, Sara, y a mis amigos de toda la vida por todo vuestro cariño y amistad desde siempre. La vida compartida con vosotros vale mucho más.

Gonzalo Varela Bartrina

Enero, 2012

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

In recent years, visual surveillance systems have been placed all over the world due to the increasing people's concern about security and the new technological developments in this area. Nowadays, cameras can be found everywhere, both in public areas such as airports, stations, city streets; and private areas like industrial spaces, shops or even inside the homes. Consequently, a huge amount of information is recorded everyday

These systems are often monitored by human beings, generating several disadvantages related to effectiveness and economic problems. First, the enormous amount of data generated by the surveillance systems makes impossible an effective control of the filmed data without investing an enormous amount of money in human resources. Secondly, a non-automatic monitoring implies to recheck the contents of the videos if new information is needed. For these reasons, automatic video systems and applications are emerging as a solution to improve the quality and opportunities of surveillance systems.

However, there are some common difficulties when working with surveillance videos such as illumination changes, presence of noise and other disturbances related to the poor quality of the videos. Additionally, other problems associated to video surveillance like occlusions can reduce the performance of the system.

Many different approaches have been developed to extract movement and behaviour information through trajectory analysis. The information contained in the trajectories, which includes both spatial and temporal data, is widely used in video surveillance. A large number of promising applications have been developed in the area of automated surveillance systems, including crowd statistical analysis, anomalous event detection, traffic monitoring or human behaviour identification. Due to the dependency to object trajectories of many of these systems, it is very important to perform an accurate detection and representation of the trajectories of the objects that appear in the videos. Therefore, improving trajectory extraction and representation is necessary to improve the quality and performance of many related applications.

1.2 Objectives and approach

Our system aims to obtain an accurate and compact representation for the trajectories of all the moving objects which appear in surveillance videos used as inputs to the system. Furthermore, we aim to develop a complete system capable to overcome with the most common challenges in video surveillance. The specific objectives of the research are:

- To find and study the viability of the actual solutions in the literature to handle occlusions, noise and disturbances in trajectories and other challenges related to trajectories such as dimensionality reduction or length normalization.
- To implement a method for detecting important temporal and spatial changes, making possible to work independently with the subpaths, facilitating trajectory representation.
- To find and implement a set of features that represents with a small error the paths of the objects, focusing on accuracy and complexity.
- To evaluate the performance and limitations of each of the implemented techniques.
- To evaluate the performance of the entire proposed system using surveillance video datasets.

1.3 Report structure

This report is structured as follows:

- Chapter 2: the most common problems and characteristics of video surveillance systems are studied in this chapter. Furthermore, a literature review of the current state of art is presented. This review includes some of the existing techniques in motion analysis, trajectory enhance techniques, occlusion handling and trajectory representation.
- Chapter 3: this chapter introduces the proposed system, presenting each step separately from the motion analysis to the evaluation step. Moreover, the selected algorithms and techniques will be further explained, describing the advantages and disadvantages of the proposed implementation against the state of art. Limitations and challenges found in the system implementation are discussed as well, explaining for each case the adopted solution.
- Chapter 4: this chapter shows the results for all the experiments carried out during the master thesis as well as a complete analysis of the achievements and limitations of the proposed system. The datasets and the ground truth used to evaluate the proposed system are also presented in this chapter.
- Chapter 5: conclusions and future work are presented in detail.

2

Literature Review

2.1 Principal challenges in video surveillance systems

Nowadays, surveillance systems can be found in public streets, shops, industrial plants, airports or even inside vehicles such as trains or buses. CCTV systems have rise as a crime prevention and reduction method in many countries. Additionally, these systems can provide evidences for criminal investigations and court purposes. The effectiveness of implemented CCTV control rooms has been studied by Gill et al [3], concluding that CCTV cameras only effective in reducing certain types of crimes while only displacing the rest. According to the authors, the principal consequence of the installation of these kind of systems is the decrease in the *'fear of the crime'* among the public. The study concludes that there is a need of control support to match the results that were expected. Due to the cost of maintaining a huge number of operators, the solution goes through the automation of surveillance systems. Moreover, improving these systems should be a priority, moving from the operator-controlled systems to smart video surveillance systems capable of performing high level tasks. Nowadays, the trend is to increase the independence of the systems and reduce the level of human intervention [4].
Changes in technology have made possible the development and integration of new technologies and applications such as motion detection, face recognition, person and vehicle tracking or crowd analysis. However, implemented devices are often affected by poor quality, caused by economic and practical problems. As a consequence, the cost and complexity of developing these technologies is highly increased. The problems associated to surveillance systems can be related to the quality and performance of the devices and systems, or to external factors like environmental conditions.

2.1.1 Problems related to devices and data acquisition

Several authors have studied the most common problems related to the devices used in video surveillance systems. Most of them are a consequence of low quality equipments. Some others are caused by the techniques used for data compression and acquisition. In addition, a bad choice of the position and coverage area of the cameras could lead to poor results. The most important problems are listed below [3, 5, 6]:

- Problems related with density, coverage area and positioning.

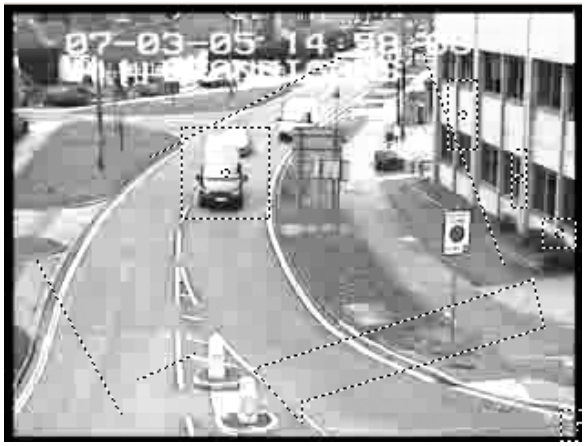

Figure 2.1: Example of false alarms caused by a sudden illumination change

- Low quality recording equipment.
- Lack of contrast.
- Noise and disturbances.
- Blurring caused by motion or lack of focus.
- Geometric distortions (severely limiting the reconstruction. of the dimensions of the objects inside the image).
- Excessive digital video compression.

Additionally, other factors such as incorrect configuration of the devices or poor camera location or insufficient disk space can decrease as well the final performance of the system.

2.1.2 Environmental and external challenges

Environmental and external factors can affect the system, reducing the final quality of the video and causing false alarms [1]. As a consequence of the enormous amount of data analysed, surveillance systems must be robust against the typical events that can produce false alarms, facilitating further work. Figure 2.1 shows an example of false alarms generated by a sudden illumination change that causes the three false objects detected in the building's wall. Muller-Schneiders et al provide some examples of the most common events that cause false alarms in [7]:

- Moving trees
- Rain
- Camera motion
- Varying illumination conditions

[1] False alarms are events identified as unusual events that do not imply any real danger or worries

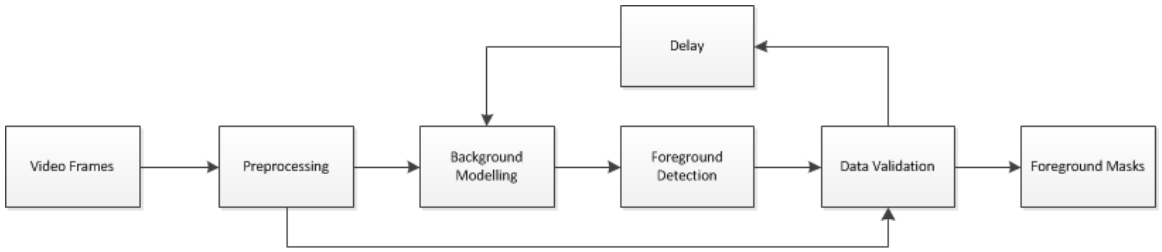

Figure 2.2: Classic structure of a background subtraction algorithm

2.2 Motion Analysis Component

The primary task of video surveillance systems is the identification of moving objects. Surveillance systems must be capable of detect and separate target objects (frequently vehicles and pedestrians) from the inanimate objects and the environment. Most of the systems use a combination of spatial segmentation and temporal segmentation (tracking) to extract the objects. The task of the spatial segmentation algorithm is to discriminate between the pixels belonging to the objects and the rest of the image. Usually, pixels belonging to the moving objects are known as foreground while the rest are considered as background. Therefore, the spatial segmentation algorithm classifies every pixel of every frame in the video as foreground or background. The next step is to find the relation of the pixels between frames, establishing which pixels belong to the same object from frame to frame. This is known as object tracking. The principal segmentation and tracking techniques are presented in the following paragraphs.

2.2.1 Spatial segmentation

In the case of static cameras (the case of our databases), the spatial segmentation usually consists in a background subtraction algorithm. Background subtraction algorithms extract the relevant information from the video (foreground), removing the quasi-constant background and reducing the amount of resources necessary to analyse the video. Background subtraction techniques are based in the generation of a model to represent the background. Each frame of the video with be compared against the simulated model to extract the foreground. Therefore, the foreground is the sum of all the detected pixels that have been found to be different from the implemented background model.
The most common problems that can affect to the performance of the background subtraction algorithm are related to the maintenance of the background model and real-time needs [8, 9]. In the following paragraphs, the most common problems that affect to the background maintenance will be presented:

- Moved objects: sometimes background objects are moved by external factors like wind in the case of the trees. These objects should not be consider part of the foreground.
- Time of the day: gradual changes in ambient illumination alter the appearance of the background
- Light switch: sudden changes in illumination due to light switching (indoor scenes) or moving clouds (outdoor scenes) change the appearance of the background.
- Camouflage: some objects pixels characteristics may be similar to the background.
- Bootstrapping: some background techniques require a training period that may be not available in some environments, thus generating a background model influenced by foreground information.

- Sleeping person: an object that became motionless sometimes cannot be distinguished of the background after a period of time.
- Waking person: when an object initially in the background moves, both it and the newly revealed parts of the background appear to change.
- Shadows: foreground objects often cast shadows that should be ignored as part of the background.

According to [9], background subtraction algorithms normally consist of four steps: preprocessing, background modelling, foreground detection and data validation.

1. Preprocessing: this step consist of a number of tasks to change the video settings, being the objective obtain better results in further steps. The most common operations include smoothing (to reduce noise), frame size or rate reduction and data format conversion.
2. Background Modelling: generation of a background model. Actual approaches can be classified in two groups: non-recursive and recursive [9]. The principal difference between the two approaches is how they upgrade the background model. The non-recursive techniques use a sliding-window that only takes in account several frames, ignoring the past story of the frames; recursive techniques upgrade the model with every input, allowing past frames to affect the current model. Non recursive models have the advantage of being highly adaptive while recursive techniques can generate more accurate models using pixel's history.
3. Foreground detection: there are two common approaches to foreground detection: estimating the absolute difference between the input pixels and the background model or using a threshold based on normalized statistics.
4. Data validation: this step consists in improving the foreground using information obtained outside the background model. There are three typical problems when using a background subtraction algorithm:
    - Lack of information of the pixels neighbours correlation: the result of this problem is the appearance of small false or positive regions.
    - Incorrect rate of adaption: the rate of adaption does not match the moving speed of the foreground objects. This can result in the appearance of ghosts[2]. Some common solutions include running several model at different rates and colour segmentation.
    - Shadow cast: caused by moving objects. A comparison of shadow removal algorithms can be found in [10].

2.2.2 Temporal segmentation

Temporal segmentation or tracking is the following step after spatial segmentation. The frames belonging to the foreground have been extracted, but it is necessary to find the relation of these pixels between the frames. The aim of object tracking is to find these relations, allowing the obtaining of high level concepts like trajectory. Tracking algorithms can be classified depending on the features used to match the blobs. In [11] the authors propose a classification based on three concepts: point tracking, kernel tracking and silhouette tracking. In the next paragraphs, these approaches will be further explained:

[2]Ghosts are large areas of false foreground

- Points: The detected objects are represented by points. The association of the points is done with the information of the previous object state. This method requires an external detector to detect the objects in every frame. This techniques can be divided in deterministic models and statistical models.
  - Deterministic models: deterministic methods define a cost of associating each object in frame $t-1$ to a single object in frame t using a set of motion constraints. Minimization of the cost is formulated as a combinatorial optimization problem. Used constraints include proximity, maximum speed or common movement.
  - Statistical models: use the state space approach to model the object properties such as position, velocity, and acceleration. These measurements are used to predict object's position. Some of the most important statistical methods include Kalman filter, particles filter or multiple hypotheses tracking (MHT).
- Kernel: kernel is related to the objects shape and appearance. Objects are tracked by calculating the motion of the kernel in consecutive frames. These techniques can be divided in two groups:
  - Template and density-based techniques: these techniques perform the matching by comparing a template or a defined geometric shape of the detected foreground in the current frame. Widely used due to their low computational cost.
  - Multi-view techniques: use different views of an object to perform the tracking.
- Silhouette: these algorithms estimate the object region in each frame. Silhouette techniques can be classified in two subgroups:
  - Contour evolution: iteratively evolves an initial contour adapting to the new object's position.
  - Shape matching: looks for a concrete shape in each frame by comparing the existing ones with the modelled shape. This model is upgraded regularly.

2.3 Occlusion handling

One of the main challenges in video surveillance systems is the problem of occlusions in video sequences. Occlusion occurs when an object is not visible in a frame because other object or static structure is blocking its view. The problem of occlusion is normally treated as a tracking issue, as position and velocity of an occluded object are difficult to determine. However, some authors propose parallel or post-tracking occlusion handling techniques in the case the tracking algorithm is not robust enough or cannot deal with occlusions [12, 13]. Javed and Shah difference three types of occlusions [1]:

- Inter-object occlusion: an object blocks the view of other objects in the visual field of the camera. An special case in inter-object occlusion is when objects are entering/leaving the filmed area. Since people usually move in groups, which results in frequent inter-object occlusion so detecting and resolving inter-object occlusion is important for surveillance applications. Moreover, traffic monitoring applications must deal also with lots of inter-object occlusions.
- Occlusion of objects due to thin structures: poles or trees can split the detected object in two different regions. This problem is aggravated when multiple objects are occluded by the same structure.

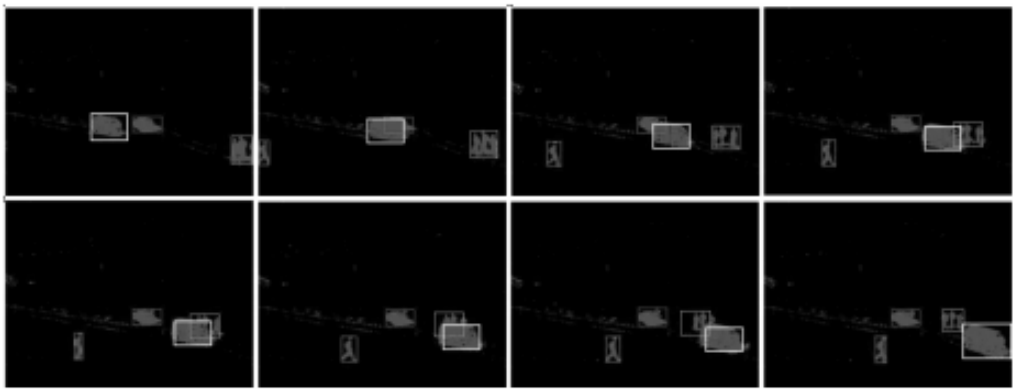

Figure 2.3: Example of occlusion provided by [1]

- Occlusion of objects due to large structures: the object disappears for a certain amount of time. It is necessary to determine the possibility of reappearance and exit of the scene. This type of occlusion is quite common in videos filmed in outdoor areas such as streets or parks.

Other proposed occlusion classifications based on tracking include additional types such as apparent occlusions or self-occlusions [14]. A brief study of the some of the most common approaches to occlusion solving can be found in [15]. All the techniques mentioned in the work are tracking approaches, the most relevant are the following:

- Usage of a ground plane representation combined with an estimation of the object size.
- The addition of an extra pixel model characterizing occlusion pixels, hence, using three different models: foreground pixels, background pixels and foreground occluding pixels.
- Learning a model of scene occlusions from the track of moving agents using minimum description length. This model creates successively more detailed models by dividing the scene into layers.
- Modelling of occlusion location based on a classification of long-term, short-term and border occlusions.

Scene modelling can be useful for occlusion handling (specially with entering/leaving objects) and can be implemented along with the occlusion prevention algorithm. In [16] Stauffer describes a method based on modelling entrances and exits with Gaussian Mixture Models (GMM).The techniques based on post-tracking processing must resolve the problem of the missing information. Ivanov et al [12] use an interpolation algorithm to fill in the blank space.

2.4 Trajectory processing

As mentioned in the section Principal challenges in video surveillance systems, there are several problems to deal with in an video surveillance system. One of the main problems, occlusion, was explained in the section before along with the most representative techniques used to solve it. However, most of the effort for trajectory representation and classification is spent in several

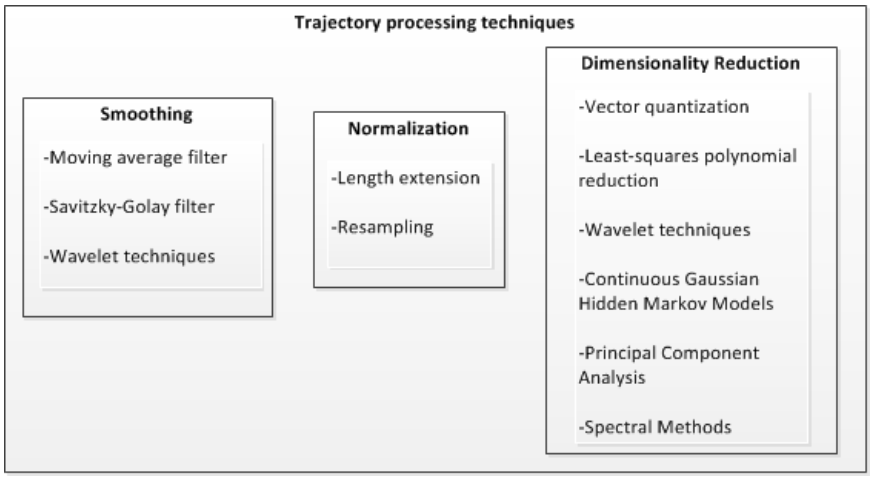

Figure 2.4: Summary of the principal techniques in trajectory processing

different operations to adapt raw trajectories to a more desirable form. Usually, two different problems are taken into account: excessive spatial variation due to noise and issues related to the tracker technique; and unequal length caused by the time-varying nature of the trajectories. The first problem affects directly to the quality of the detected paths and complicates further operations. One of the most common approaches to solve this problem is data smoothing. The second problem must be solved to allow the application of clustering techniques, which normally require equal length in input data. Most researches use a combination of trajectory normalization and dimensionality reduction in order to prepare raw trajectories for clustering.

2.4.1 Data smoothing

Noise and disturbances affect directly to the final efficiency and performance of the video surveillance systems. Moreover, noise contamination is a common problem in many digital systems, specially when the quality of the input is questionable. Data smoothing is a wide used technique in lots of different fields to reduce the effect of noise on input data. Additionally, this solution can help to eliminate little irregularities in the captured data, providing a clearer view on the behaviour of the trajectories. There are lots of different approaches to data smoothing depending on the needs of the particular field or application of study. Some of the most representative techniques applied to surveillance systems are mentioned in [17]:

- Moving average filter: is a type of finite impulse response filter commonly used to analyse a set of points by creating a series of averages of different subsets of the full data set. Moving average techniques are commonly used with time series data to smooth out short-term variations and highlight longer-term trends.
- Savitzky-Golay filter: performs a local polynomial regression on a series of values to determine the smoothed value for each point. Savitzky-Golay filter is used for smoothing and differentiation in many fields and has been well studied in literature [18].
- Wavelets: this technique is based in the decomposition of the signal in two: a high frequency and a low frequency signal. The low frequency signal preserves the main structure characteristics of the original signal. Thus, the low frequency signal can be viewed as a smoothed version of the original signal.
- Other low-pass filters: most of the filters used in data smoothing are low-pass, including some of the mentioned above.

2.4.2 Normalization

Normalization consists in ensure that all the trajectories have the same length, which is a necessary condition to further clustering. Normally, the chosen length is the length of the longest trajectory. There are two different approaches: length extension and resampling. In [17], these two techniques are further explained:

- Length extension: consists in the addition of a number of new samples to reach the required length. The two most common approaches are zero padding and track extension. Zero padding is based on add extra zero samples concatenated at the end of the trajectory. Track extension consists in estimate the value of the new samples using dynamic information of the trajectory.
- Resampling: uses interpolation to guarantee that all the trajectories have the same length. The most popular technique is linear interpolation. If sample reduction is necessary, then subsampling is the best choice.

2.4.3 Dimensionality reduction

Dimensionality reduction techniques are widely used in many fields to reduce the dimensionality of the original data prior to any modelling of the data. In trajectory processing, these techniques are used to map trajectories to a more computationally manageable space. The new space should be chosen using the parameters that best characterize the trajectory model. Morris and Trivedi [17], describe some of the most important methods:

- Vector quantization: based on limit the number of possible trajectories to a finite alphabet of prototypical vectors.
- Least-squares polynomial reduction: approximation of the trajectory based on least squares method, modelling the trajectory as a 2-D curve.
- Wavelet techniques: representation of a trajectory at different levels of resolution.
- Continuous Gaussian emission hidden Markov models: used to characterize the trajectory by its temporal dependencies between samples and then use the parameters obtained to represent the trajectory with lower dimensionality.
- Principal component analysis (PCA): a popular method based on an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.
- Spectral methods: uses eigenvalues obtained from a matrix constructed with the training trajectory set and then a transformation to the spectral space to obtain the new low dimensionality trajectories.

2.5 Trajectory segmentation

Trajectories extracted from moving objects in videos are normally used to represent the objects movement in surveillance systems. However, object movement can be complex and the trajectory can be divided in subpaths, each one containing a fraction of the total information. Every point of the original path must belong to one the subpaths. Thus, rejoining the subpaths in

chronological order must result in obtaining the original trajectory. The criteria used to divide the trajectory can be of spatial or temporal nature. The segmentation algorithm detects the points where there is a change in the motion such as a direction change or a speed increment. Consequently, each subpath represents a portion of the trajectory in which motion features remain constant or quasi-constant.
The main advantage of trajectory segmentation is the simplification of further analysis, using subpaths instead of the complete trajectory. Moreover, as each subpath represents a concrete part of the total movement, is easier to extract and isolate particular behaviours. This information can be used in combination with the analysis of the complete trajectory, obtaining a more robust model. Trajectory segmentation can be also used to obtain a reduced dimensionality model, using the transition points between subpaths as the new trajectory points. Although this simplification means that the original points cannot be recovered, interpolation can be used if is necessary to increment the number of points of the trajectory. The procedure to detect the points where there is a significant change in the spatio-temporal characteristics of the trajectory, known as breakpoints, is normally based on geometrical and temporal analysis. Despite some of the surveillance systems and applications based on trajectories do not use a segmentation phase, several authors include this step in their systems with promising results. Piotto et al [2] propose a system based in three different algorithms. After the segmentation, breakpoints are characterized by a set of features and then mapped to a syntactic alphabet. This leads to a simple matching step that can be used in high level applications such as automatic detection of anomalous events. The authors classify the breakpoints in two types:

- Temporal breakpoints: detect sharp velocity discontinuities in the object motion. In particular, this breakpoints are useful to detect stops/re-start events in the trajectory.
- Spatial breakpoints: denote changes in the direction of the object. In this category, is necessary to distinguish between short-term variations and long-term variations using a different algorithm for each type.

Bashir et al [19] use curvature changes in trajectory to segment the paths, and then PCA to represent the subpaths. The subpaths are grouped using clustering and then a HMM representation to build a motion recognition system.

2.6 Trajectory representation

Trajectory representation techniques are often used in high level applications to facilitate procedures like clustering and matching. There are several approaches to obtain a more robust and efficient model. Moreover, a good representation strategy should be able to highlight similarities and differences between trajectories, as same spatial trajectories could be associated to different motion patterns. Basically, there are two approaches to trajectory representation. On the first hand, some authors use a set of features to represent the main characteristics of the trajectories, applying in some cases a transformation to the obtained features to improve robustness or reduce dimensionality [20, 12, 2]. On the other hand, some others, use direct transformations on trajectories such as PCA [19]. Porikli and Haga propose a feature classification in [20]. This classification divides the features in two groups: object-based features and frame-based features. The first type describes properties of individual objects while the latter represents the properties of a frame using the properties of objects existing in the frame. The main features described in each group are the following:

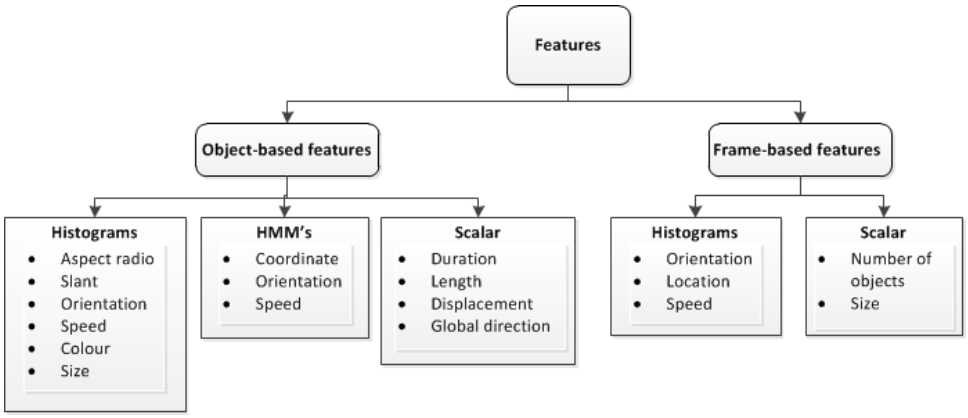

Figure 2.5: Summary of the principal features used in trajectory representation

- Object-based features
  - Histograms: aspect radio, slant, orientation, speed, colour, size.
  - HMM's: features transferred to a parameter space λ that is characterized by a set of HMM parameters. This type includes: coordinate, orientation, speed.
  - Scalar: duration, length, displacement, global direction.
- Frames-based features
  - Histograms: orientation, location, speed.
  - Scalar: number of objects, size.

To reduce the high dimensionality of most of the proposed features, the authors use an eigenvector decomposition on the similarity feature matrices. Recent work with this type of features has been developed by several authors. The following paragraph describes some of these approaches: Ivanov et al [12] propose a representation based on model every trajectory with high level features such as velocity and acceleration. These features are re-sampled to ensure the same length for all the trajectories and then fed to a SVM system for training. In [2], the authors propose also a system based in features extraction. Nevertheless, features extraction is done for every breakpoint calculated in a previous step. The features used are angle, velocity and duration. Then, these parameters are quantized and prepared for the matching procedures.
Bashir et al [19] estimate the PCA coefficients of the previously obtained subpaths to represent the trajectories in an accurate and compact way. To achieve invariance, input trajectories are first low-pass filtered and normalized. Other common approaches to dimensionality reduction used in trajectory representation can be found in Section 2.4.3.

3

System design and development

The aim of this chapter is to present the different functional stages in which the framework is composed of, and provide the reasons to justify the technologies used. The chapter is subdivided following the structure of the system.

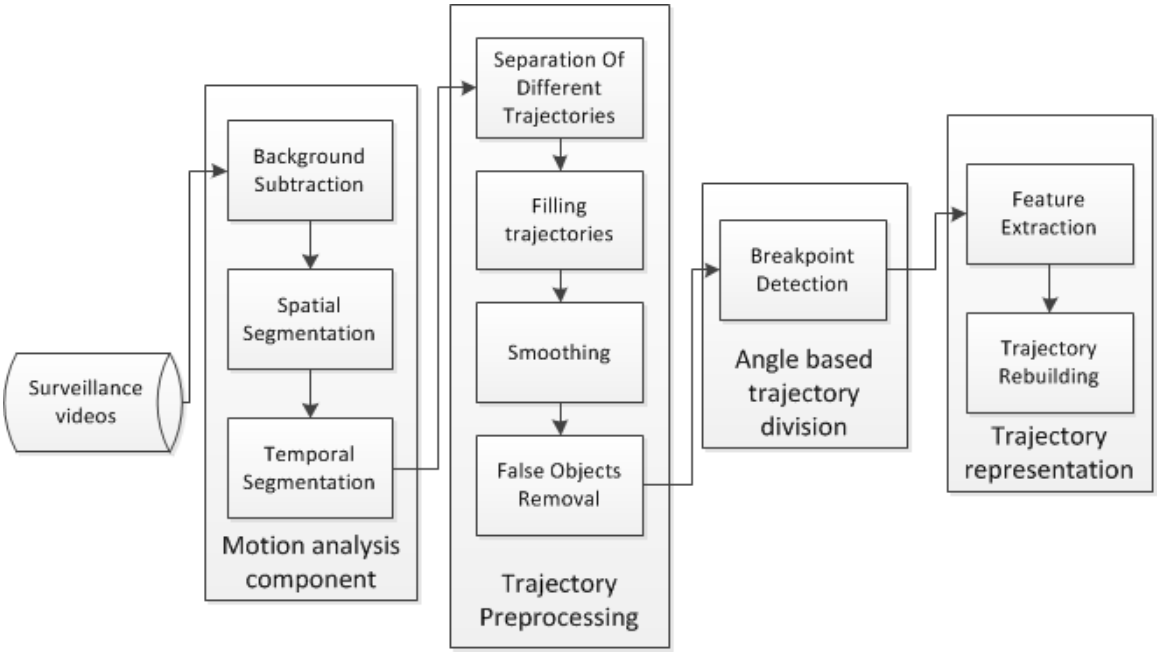

Figure 3.1: System structure

The main objectives of the system are:

- Extract trajectories of moving objects from surveillance videos.
- Refine the raw trajectories to facilitate further work.
- Find a suitable model for the representation of the refined trajectories.

The project is divided in four main stages: motion analysis component, trajectory preprocessing, angle-based trajectory division and trajectory representation. All the proposed techniques and algorithms have been implemented in C/C++.

1. **Motion Analysis Component** (section 3.1) receives the surveillance video and detects the moving objects in the video, providing the raw trajectories that will be further analysed in the next steps. The objects in the video are spatially and temporally segmented to remove the non-relevant information, basically the parts of the frames not belonging to the objects, therefore reducing the amount of the information that will be analysed. Thus, the motion analysis component yields a background subtraction method, an object segmentation algorithm and an object tracker. The information provided includes the size, coordinates of the centroid and frame numbers where the detected objects appear.

2. **Trajectory preprocessing** stage (section 3.2) although the motion analysis component is based on a robust technique, some matters like noise and occlusions can lead to a more difficult and less accurate results in the next steps. Consequently, is essential to enhance this data. Several operations are defined in this step: separation of mismatched trajectories[1], joining the subpaths of a trajectory split by occlusions, a smooth step based in Saviztky-Golay filters and a final false object[2] removal step.

3. **Angle based trajectory division** stage (section 3.3) consists in the separation of each trajectory into several subpaths. Trajectories are scanned to find the points where there is an important spatial or temporal change, that is, the breakpoints.

4. **Trajectory representation** (section 3.4) once the breakpoints have been calculated, the next step is to extract the features that characterize the subpaths existing between breakpoints. Finally, these features can be used to reconstruct the subpaths. The aim of this stage is to procure a method to model and represent trajectories which have been previously pre-processed.

In the following paragraphs each of the presented stages will be further detailed along with the techniques applied in each stage.

3.1 Motion Analysis Component

The first stage of our framework is the Motion Analysis Component. Its objective is the extraction of the moving objects that appear in the surveillance videos. Relevant information is contained in the movement objects such as vehicles or pedestrians, while the background remains quasi-constant in time. Therefore, it is necessary to separate the foreground objects from the background to allow the analysis of this information. Moreover, this process leads to low computational cost and faster procedures, reducing the amount of information processed. However, some operations like resizing and changing the input video format are necessary before starting the analysis. The stages that compose the Motion Analysis Component are the following: preprocessing operations, background subtraction algorithm, spatial segmentation and object tracking.

[1] Mismatched trajectories are trajectories that belong to different objects but have been merged by the tracker algorithm as if they belong to the same object.

[2] False objects are detected objects not extracted from the detection of a real object but created by noise or other disturbances.

- Input preprocessing step grants that the input video will match the system requirements of size and format. A video edition tool is used to change the video parameters.
- Background modelling is necessary to separate foreground objects from the background. The technique used is based in modelling the background as a mixture of Gaussians.
- Object detection step finds out the spatial nexus of the detected foreground pixels in each frame. The method used is a two-pass connected component algorithm based on a 8-neighbour connection.
- Object tracking provides the relation of the objects detected between different frames. The implemented solution is based on a linearly predictive multi-hypothesis tracking algorithm.

The Motion Analysis Component fundamentals and steps are explained in further detail in the next paragraphs.

3.1.1 Background subtraction algorithm

In this step,the objective is to calculate a mathematical background model, based in the work of Stauffer and Grimson [21]. This approach provides an adaptive background subtraction technique that deals robustly with illumination changes, slow moving objects and other external problems that affect typically to background subtraction procedures. This method is also suitable for working on real-time applications using low resolutions. In the approach of Stauffer and Grimson [21], the separation between the foreground of an image and the background is performed by modelling the background as a mixture of Gaussians. Stationary objects are considered as part of the background, while moving objects are subscribed to the foreground.
For a particular pixel, the probability of belonging to the background or the foreground is calculated based on the persistence and the variance of each of the Gaussians of the mixture. Furthermore, this value is directly related to the intensity and colour distribution of the pixel in that certain image. As a result of this method, each pixel is classified, considering foreground pixels those which do not fit any background distribution. Due to possible light changing conditions, the model must be based in adaptive Gaussians. The main advantages of adaptive Gaussians are: the robustness against external changes and the capacity of deal with sleep walking objects. The later advantage comes from the fact that the previous background still exists but with a lower ω.
The recent history of each pixel is modelled as a mixture of K Gaussian distributions. The probability of observing the current pixel value is:

$$P\left(X_t\right) = \sum_{i=1}^{K} \omega_{i,t} * \eta\left(X_t, \mu_{i,t}, \epsilon_{i,t}\right) \tag{3.1}$$

where K is the number of distributions; $\omega_{i,t}$ is an estimate of the weight of the i^{th} Gaussian in the mixture at time t; $\mu_{i,t}$ and $\epsilon_{i,t}$ are the mean value and covariance matrix of the i^{th} Gaussian in the mixture at time t; and η is a Gaussian probability density function. Every time a new pixel value X_t is obtained, this value is checked against the existing K Gaussian distributions until a match is found. In this process, the authors define match as a pixel value within 2.5 standard deviations of the checked distribution. Nevertheless, in the case of mismatch, the least probable distribution is replaced with a new distribution, using the current value as its mean value, and an initially high variance and low prior weight.
Once the pixel has been classified its model needs to be adjusted, thus, the weight, average and

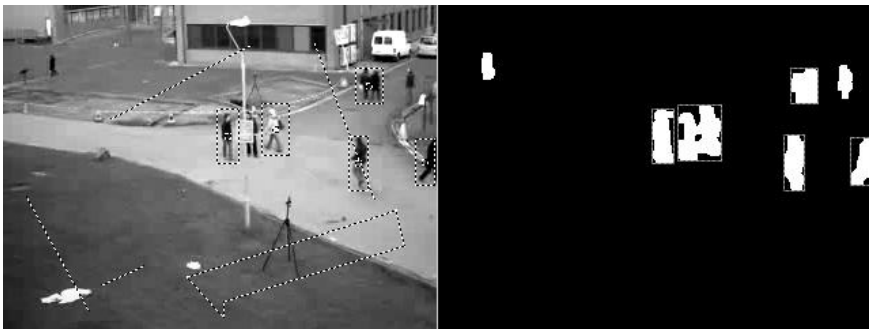

Figure 3.2: Object segmentation method example

variance are updated following the formulae 3.2, 3.3, 3.4 and 3.5.

$$\omega_{k,t} = (1 - \alpha)\,\omega_{k,t-1} + \alpha\,(M_{k,t}) \tag{3.2}$$

where α is the learning rate and $M_{k,t}$ is the result of the pixel classification (1 when the model is matched and 0 otherwise).

$$\mu_t = (1 - \rho)\,\mu_{t-1} + \rho X_t \tag{3.3}$$

$$\sigma_t^2 = (1 - \rho)\,\sigma_{t-1}^2 + \rho\,(X_t - \mu_t)^T\,(X_t - \mu_t) \tag{3.4}$$

where ρ

$$\rho = \alpha\eta\,(X_t|\mu_k, \sigma_k) \tag{3.5}$$

In the case of mismatch, only the weights are updated.

3.1.2 Spatial segmentation

After the pixels are classified, foreground pixels are grouped using a two-pass connected components algorithm. The algorithm scans the image until it finds a pixel classified as foreground. Each pixel has a label attached to it; zero if belongs to the background and an integer otherwise. Then, basing on the label of the four neighbours that have been already scanned the pixel is updated as foreground or background. After the first pass, the scan is repeated again because the connected components algorithm is two-pass. Furthermore, each label is associated with a class, in the form of an identification number. This number is replaced by its equivalent label during the second scan. Hence, foreground objects are segmented spatially in every frame of the video.

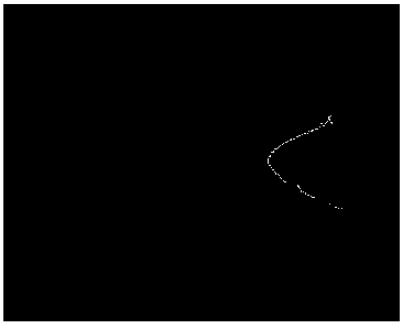

Figure 3.3: Example of raw trajectory

3.1.3 Multiple hypotheses object tracking algorithm

Once the objects have been spatially segmented, the next step is to establish the correspondence of these objects between frames, that is, to initiate temporal segmentation. This process is accomplished using a linearly predictive multiple hypotheses tracking algorithm. Each analysed frame has an associated pool of Kalman models, obtained using Kalman filters. The models are probabilistically matched to the connected components that they could explain. Each match with a small error is used to update the model. These models will be used in the following frame. If no match is found for a particular model, this model will still be propagated but the probability of being used will be decreased. The unmatched models from the current frame and the two previous ones are used to hypothesize new models, basing on pairs of unmatched components. If the current frame contains a match with small error, the model is added. Finally, least probable models are removed if excessive models exist.

3.1.4 Summary

In conclusion, the motion analysis component consists in three different operations: (i) background modelling to remove video redundant information (which is mainly the scene background); (ii) object detection using connected component analysis to group the extracted foreground pixels; (iii) object tracking based on Kalman filters.
The analysis of the trajectories extracted in this component is discussed in the subsequent chapters. An example of these raw trajectories can be found in Figure 3.3

3.2 Trajectory preprocessing

The raw trajectories provided by the previous component (Section 3.1) are refined in this stage, in order to facilitate and improve the results for further operations. The main challenges to deal with are: incorrect assignment between trajectories and objects produced by crossings and occlusions, and distortion and imperfections caused by noise. The former difficulty can cause two types of confusions: the first, assigning different trajectories to the same object; and the second one, assign different parts of the same trajectory to different objects. The latter, increases the complexity of the trajectories, thus, increasing the difficulty of further stages. Moreover, noise related to illumination changes and reflections in objects and surfaces can introduce false objects in the system. Several solutions have been implemented to deal these challenges:

1. **Separation of objects with different trajectories** is the first stage of the trajectory preprocessing component. To deal with the matter of different trajectories assigned to the same object, raw trajectories are analysed and split if the one of the conditions (further explained in Subsection 3.2.1) is fulfilled.

2. **Joining of trajectories that belong to the same object**. The main cause of the division of some of the trajectories in the videos into several subpaths are occlusions and crossings caused by obstacles and other objects. The technique proposed to solve this is based in Bresenham's line drawing algorithm (see subsection 3.2.2 for more details).

3. **Trajectory smoothing** is the third stage of the trajectory preprocessing component. The aim of this stage is reduce the noise and imperfection of the raw trajectories. The method implemented is based in Saviztky-Golay filter (details in subsection 3.2.3).

4. **False objects removal** is the last stage. In order to eliminate the false objects introduced by noise effects, the objects that have been smoothed have to fulfil a number of conditions (details in subsection 3.2.4).

3.2.1 Separation of objects with different trajectories

Sometimes, when two objects pass near each other or their paths cross, the motion analysis component assigns the paths as belonging to the same object. Therefore, it is necessary to analyse raw trajectories to find and separate the different paths. A set of conditions is defined to determine when a trajectory should be divided. These conditions are strictly spatial, no temporal information is considered, because the paths involved are always consecutive in time. The conditions are the following:

1. **Distance between two consecutive points:** the distance is defined as the Euclidean distance:

$$\epsilon_n = \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2} \tag{3.6}$$

   If ϵ_n exceeds a pre-defined threshold ϵ the condition is satisfied and the object divided.

2. **Angle** β (defined in *Beta breakpoints* 3.3.2) condition is satisfied if the angle exceeds the threshold α_1. The election of the threshold is based on the idea that moving objects extracted from surveillance videos rarely change abruptly their direction.

3.2.2 Joining of trajectories that belong to the same object

Occlusion is one of the main problems when working with video surveillance videos. Pedestrians and cars often get occluded by a number of static obstacles like traffic lights, trees or posting signs. In addition to these obstacles, other targets can produce the same effect when passing over the focused object. As a result, the trajectories are split into several others, not representing the real behaviour of the object. To detect occlusions, the system compares each trajectory with all the others, and if they satisfy a number of conditions, the trajectories are joined:

- **Spatio-temporal proximity:** the distance between the last point of the first path P^1_{end}(the one that goes before in time) and the first point of the second path P^2_{beg}, must

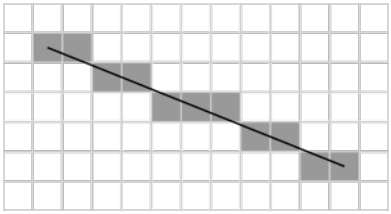

Figure 3.4: Bresenham's algorithm

be under than a predefined number N (the measure of distance is the same as in Formula 3.6). In addition to this, the difference between the number of the frame of P^1_{end} and P^2_{beg} cannot exceed F. This conditions ensures that only spatio-temporal close paths can be joined.

- **Similar direction:** this condition is similar to the beta angle condition of the previous stage, but in an opposite way. The end of the first path and the beginning of the second one are analysed using the same formula as in 3.2.1. The condition is satisfied if the value obtained is below a predefined threshold α_2. This condition prevent the system from joining paths of different crossing objects that would probably satisfy the rest of conditions.

- **Consecutive paths:** to ensure the second path is consecutive in time, the number of frame in which appears the last point of the first path (F^1_{end}) must be smaller than the number of frame of the first point of the second path (F^2_{beg}). Non temporally consecutive paths cannot belong to the same object.

- **Similar size:** for each path, an average size is calculated. A maximum increment ΔS is defined, discarding the compared paths that overcome that value. This condition avoids the system from joining paths from different objects that have satisfied the spatio-temporal proximity condition ignoring the effect of the image depth. This effect can be detected comparing the size of the objects. Additionally, can help to avoid joining different kind of objects, for example pedestrians and cars.

Whenever two trajectories are joined, there is always an empty gap between them. The missing trajectory segment is completed using the filling process described in [12]. This process is based in the Bresenham's line drawing algorithm [22], whose operation principles are fully detailed the next paragraphs.

Bresenham's drawing algorithm

The Bresenham drawing algorithm determines which points in a n-dimensional space should be selected to form the line between two given points. The principal advantages of the algorithm are its simplicity and its low computational cost (as it uses only cheap computational operations such as integer addition, integer subtraction and bit shifting). Our implementation of Bresenham's line drawing algorithm is described in the following paragraphs.
The first stage is the initialization of the variables used by the algorithm. The first operation is to calculate the variables dx and dy:

$$dx = x_1 - x_0 \tag{3.7}$$

$$dy = y_1 - y_0 \tag{3.8}$$

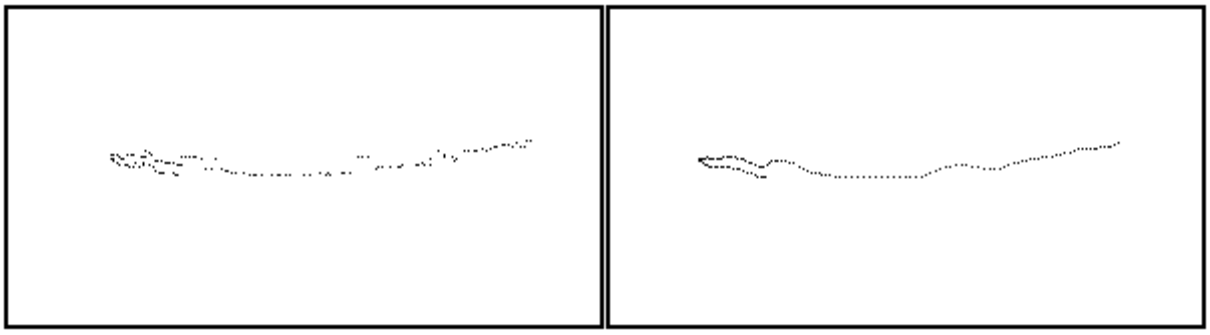

Figure 3.5: Comparison between pre-smoothing trajectory (left) and post-smoothing trajectory (right)

where (x_0, y_0) are the coordinates of the initial point and (x_1, y_1) represent the coordinates of the final point The values obtained are used to calculate inc:

$$inc = 2 * dy - dx \tag{3.9}$$

The second part of the algorithm is a loop where the stopping criterion is to reach $x = x_1$. In each iteration, the value of x is increased in a unit. The value of y can be updated depending on inc, which is also updated in each iteration. There are two different possibilities: $inc > 0$ and $inc <= 0$. If $inc > 0$, y will be updated and the current point (x, y) will be extracted. Then the variable inc will be updated with the formula $inc = inc + (2 * dy - 2 * dx)$. In the case that $inc <= 0$, y will not be updated and $inc = inc + 2 * dy$.

3.2.3 Trajectory smoothing

The object trajectories obtained after separation and filling remain noisy and imperfect. The objective of the trajectory smoothing is to reduce the effects of illumination changes, noise and other imprecisions related to external factors . Besides, smoothing reduces the complexity of the trajectories, facilitating representation as the number of features necessary to characterize them decreases. A solution to smooth the trajectories obtained is presented in [12]. Authors propose an implementation based on Saviztky-Golay filter. In the next paragraphs this filter and its implementation are further explained ir order to facilitate the comprehension of the proposed system.

Saviztky-Golay filter

Saviztky-Goaly filter is based on the calculation of a polynomial regression to determine the value of each point. The filter has three parameters: order, polynomial degree and length. The order of the filter is the order of the nth derivative that the filter would apply to the entrance (zero order is equal to smoothing). The polynomial degree is related to the polynomial regression used to calculate the values of the filter. Finally, the length of the filter is the number of points of the filter used, must be always an odd number and higher than the polynomial order. The number of points of the trajectories must be at least the length of the filter.
For a certain order, filter coefficients are dependent on the polynomial degree and length. As the polynomial degree decreases or the length increases, the noise variance is reduced, nonetheless

this implies losing some details of the input signal [18]. The main advantage of the Saviztky-Golay filter is that it tends to preserve the features of the distribution such as minima and maxima relative values. According to the results obtained by Ivanov et al [12], the selection of a proper polynomial order and window length leads to lower computational complexity and a good approximation of the trajectories. The parameters are set to the values proposed in [12]: polynomial degree $p = 4$ and length $n = 21$. Figure 3.5 shows the effects of the filtering over a raw trajectory with $p = 4$ and $n = 21$.
The steps followed for the implementation of the filter are described in the next paragraphs:

1. This first point consists in obtaining the matrix A. A is a matrix of $nx(p+1)$, where n is the length of the filter and p is the polynomial degree. This matrix is calculated $k+1$ times, where $k = \frac{n-1}{2}$, using the following formula:

$$A_m = (O_m^T * Q) \forall m = 1, 2...k+1 \tag{3.10}$$

   where O_m is a vector containing the numbers from $(1-m)$ to $(n-m)$ and Q is a vector of $(p+1)$ one values. Finally, each column of A_m is raised to the $(r-1)$ power where r is the number of that column.

2. The second step is finding A_m^+ the pseudo-inverse matrix of A_m. The method used to compute A_m^+ is the single value decomposition (SVD) [23]. The single value decomposition consists in a factorization of the form $A_m = U_m * S_m * (V_m^*)^T$, where U_n is a n x $(p+1)$ matrix, S_m is a diagonal matrix of $(p+1)$ x $(p+1)$ dimensions and V_m is a matrix of $(p+1)$ x $(p+1)$ dimensions. The implementation of the SVD method is mathematically complex, consequently, our approach is based on the implementation described by Press et al [24]. Once the U, S and V matrices are calculated, the pseudo inverse is obtained by the following operation:

$$A_m^+ = V_m * S_m^+ * U_m^T \tag{3.11}$$

   where S_m^+, the pseudo inverse of a diagonal matrix, can be calculated by taking the reciprocal of each non-zero element on the diagonal and transposing the resulting matrix. The first row of the matrix A_m^+ corresponds to the mth row of the filter, F_m.

3. The procedure presented in steps 1 and 2 is performed $k+1$ times, thus, obtaining the first $k+1$ rows of the filter. The remaining rows from $i = k+2$ to n are calculated by using $F_i = F_{n-i-1}^{inv}$, where F^{inv} is the same matrix as F but with the values of the rows sorted inversely.

3.2.4 False objects removal

False objects are created by rapid illumination changes and reflections in objects and surfaces. Additionally, little movements of parts of the background such as trees can contribute to increase this effect. All these challenges are further explained in Section 2.1.2. In order to eliminate false objects, objects are checked against a number of conditions. These conditions are the following:

- **Minimum time duration:** the object must appear more than one second

- **Minimum number of points:** the number of points of the trajectory (which is invariant during the trajectory pre-processing) must be above a predefined minimum κ. False object trajectories are normally short due to its dependence to rapid and chaotic changes in noise and illumination. Moreover, trajectories with a very short number of points provide little or none information of the behaviour of the object. Consequently, trajectories with less points than κ will be removed in this stage. κ is bounded below by the selected length of the Saviztky-Golay filter (see *Saviztky-Golay Filter* 3.2.3).
- **Minimum stroke:** the object must change its position in more than a certain number of pixels. The reason beneath this condition is that objects created by external factors are normally stationary, moving little around the same position. The stroke condition implementation consists of a minimum distance from the initial point that the object must overcome.
- **Minimum size:** the object must have a minimum size. Objects smaller than a pedestrian should not be taken into consideration. Additionally, pedestrians smaller than a certain size cannot be distinguished as a moving object. Regarding this, a size threshold is implemented. The value of the threshold is set experimentally.

3.2.5 Summary

The main objective of the presented section is to enhance the information obtained in the motion analysis component. As previously stated, the trajectory preprocessing component consists of: (i) Separation of objects with different trajectories, to prevent several trajectories to be artificially joined in one object (ii) Joining of trajectories that belong to the same object, to deal with occlusion and split paths that belong to the same object; (iii) Trajectory smoothing, to remove noise and reduce the complexity of the trajectories; and finally, (iv) False objects removal to eliminate objects artificially created by noise. After these operations, the resulting trajectories pass to the next step.

3.3 Angle-based trajectory division

The main objective of this stage is to procure a method to divide trajectories which have been previously preprocessed in several subpaths. In the videos, the objects change their movement in different ways such as stops, deviations and turns. Hence, these changes are reflected in their trajectories which evolve temporally and spatially. To represent trajectories, is easier to divide them in several subpaths, trying to separate parts of the trajectory with different behaviour. The points where trajectories change are known as breakpoints, and their detection is the first step for trajectory division. After the detection, trajectories are divided into several subpaths, each of them starting and finishing in a breakpoint. Consequently, each subpaths represents a part of the trajectory with a different behaviour compared to the ones before and after.
In our approach, we differentiate three different types of breakpoints: temporal, beta-spatial and gamma-spatial. Temporal breakpoints mark the points where the velocity of the object detected is null, which often indicates a following change in the trajectory. Moreover, stops usually are related to external factors such as traffic lights or meetings between peasants, thus, important information for behaviour understanding.
Spatial breakpoints denote changes in direction. These changes can be abrupt and local or cumulative and extended in a long period. Therefore, two different criteria are implemented to detect these deviations: beta breakpoints and gamma breakpoints. The breakpoints implemented in the system are defined in [2]. In the next paragraphs, the different types of breakpoints will be further explained.

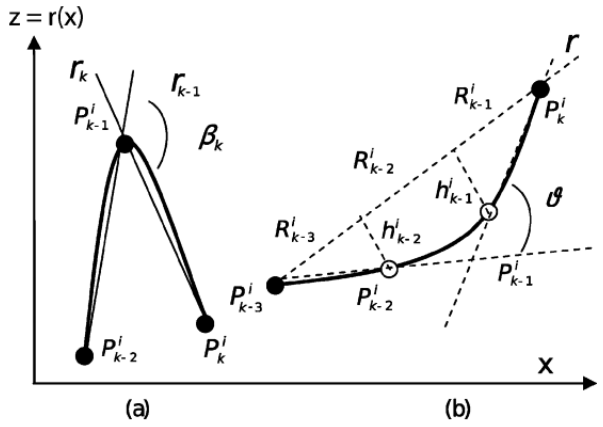

Figure 3.6: Angles used by Piotto et al [2] (a) Local variation angle (b) Long-term deviation

3.3.1 Temporal breakpoints

Temporal breakpoints mark stops in the trajectory. To detect these stops, each point is evaluated using a temporal window of length k. Furthermore, to prevent the small deviations produced by noise in the object centroid position, a guard area proportional to the object's size is established. If the following k points are inside the guard area, the point is marked as a temporal breakpoint. In the system, k is set experimentally based on the range recommended in [2].

3.3.2 Spatial breakpoints

Beta breakpoints indicate sharp deviations in the trajectory. These deviations can be caused by big direction changes (for example turning a road cross), objects dodging others or inanimate obstacles. The detection criteria is derivative, therefore useful to detect short-term abrupt changes. To find the beta breakpoints, each point of the trajectory is evaluated with the following formula:

$$\beta_k = \tan^{-1} \left| \frac{m_{k-1} - m_k}{1 - m_{k-1} m_k} \right| \tag{3.12}$$

First, the slopes (m_{k-1},m_k) belonging to the lines (P_{k-2}, P_{k-1}) and (P_{k-1}, P_k) are calculated. Then, the result of applying the formula above is compared with a predefined threshold β. If $\beta_k > \beta$, the point is marked as a beta breakpoint. Figure 3.6 shows the angle used for beta breakpoint detection.
The criterion used for beta breakpoints in this section cannot detect changes generated by successive small deviations. To detect these long-term variations, it is necessary to define a new criterion, in this case of integrative nature. The points detected with this technique are known as gamma-breakpoints. The objective is to calculate the area γ subtend by the trajectory from the last breakpoint to P_k. The integrative criterion is presented below:

$$\gamma_{(k-g,k))} = \frac{1}{2} \sum_{q=k-g}^{k} \left[\left(h_q^i + h_{q+1}^i \right) \left(R_{q+1}^i - R_q^i \right) \right] \tag{3.13}$$

As shown in Figure 3.6, r is the line that joins the last breakpoint with the point being evaluated P_k, (h_q^i is the distance between the trajectory and r, and R_{k-g}^i is the distance between the projections of the trajectory points P_{k-g} and P_{k-g+1} in r. Again, if $\gamma_j > \gamma$, the point is marked as a breakpoint. The values of β and γ are discussed in the next chapter.

3.3.3 Summary

The Angle based trajectory division stage is divided in two parts: (i) Temporal breakpoints and (ii) Spatial breakpoints. Each one is related to a different type of breakpoint, and indicates a different change detected in the trajectory. In the next section, the subpaths defined by the detected breakpoints, are characterized by a set of features (angle, velocity, duration), used to represent and rebuild the trajectories in an accurately and compact way.

3.4 Trajectory representation

According to the spatio-temporal analysis presented in the previous section *Angle based trajectory division*, the trajectory can be described as a chain of breakpoints B_n^i $\forall$ n=1,2,...M (being M the number of breakpoints for the *ith* object). Each pair of breakpoints, B_n^i, B_{n+1}^i, identifies a subpath $S_j^i = (B_j^i, B_{j+1}^i)$ that approximates a portion of the original path. Consequently, the trajectory can be represented by a description of the subpath chain. In our representation, based on [2], each subpath is characterized by three features: angle (Θ_j^i), duration (Δt_j^i) and velocity (ν_j^i). With these features it is possible to describe a wide variety of trajectories and perform different analysis. Some examples include the identification and representation of trajectories with similar spatial behaviour but different speed, detection of sleep walker objects, discriminate between cars and pedestrians based on their velocity and moving patterns and the detection of concrete behaviour patterns. Additionally, the implementation of the features is simple and compact, and the trajectory can be easily rebuilt from the feature values. In case the temporal information is not relevant, the samples with null velocity can be dropped, obtaining a pure geometric representation. Moreover, duration and velocity can be merged to form a more compact representation if looking only for spatial information.
The implemented representation is inherently invariant to rotation and translation. Only the coordinates and orientation of the first segment refer to an absolute positioning (the first point is always necessary to rebuild the trajectory), but this information can be easily discarded to achieve invariance to translation and orientation. The features are presented in the formulae below.

$$\Theta_j^i = \tan^{-1}\left(-m_j^i\right) + tan^{-1}\left(m_{j-1}^i\right) \tag{3.14}$$

$$\Delta t_m = t_j - t_{j-1} \tag{3.15}$$

$$\nu_j^i = \frac{d\left(B_j^i - B_{j-1}^i\right)}{\Delta t_j} \tag{3.16}$$

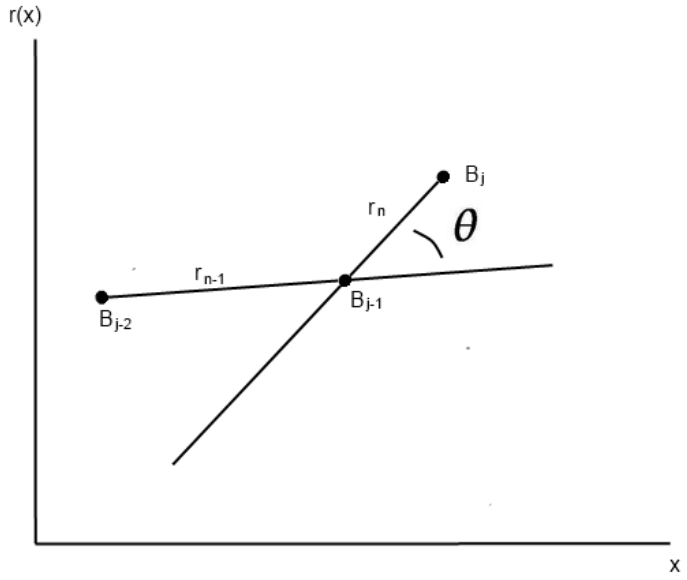

Figure 3.7: Theta angle

Direction and duration are computed with respect to the previous segment, while velocity is calculated as the length of the segment divided by its duration. Θ_j^i is calculated using the slopes m_j^i belonging to the line defined by (B_{j-1}^i, B_j^i); and m_{j-1}^i which is defined by (B_{j-2}^i, B_{j-1}^i). Figure 3.7 shows the angle Θ. In the figure, r_{n-1} is the line that connects $B_{j-2}^i and B_{j-1}^i$; r_n the one that connects $B_{j-1}^i, B_j^i.\Delta t_j$ uses t_j and t_{j-1} which are the absolute time references for B_j^i and B_{j-1}^i. Finally, in the velocity equation, $d(B_j^i - B_{j-1}^i)$ is the Euclidean distance between the two breakpoints.

In our representation, Θ angle marks, along with the distance obtained by the multiplication of velocity and duration, the position of the next point. Starting from the initial point, the system calculates the features presented above for every breakpoint detected in the *Angle Based Trajectory Division* step. Although the three features selected are the ones proposed by Piotto et al [2], the definition of Θ used in the system (see *Equation* (3.14)) is different from the proposed by the authors. The implemented definition maintains the properties of the feature set, while makes easier the trajectory rebuilding due to its simpler definition. Additionally, the velocity can be positive or negative, the sign has a purpose in trajectory rebuilding.

3.4.1 Trajectory rebuilding

The last step is to perform the rebuilding of the trajectory using the features presented above. The system starts from the first point of the trajectory, and then calculates the first breakpoint using the three feature values previously obtained. Then, starting from the obtained breakpoint, calculates the next one using the next group of features.

To calculate the next breakpoint, the system uses the slope of the prior segment m_{k-1}(except in the case of the second point, because no prior segment exists) and the angle Θ to get the slope of the new segment using the formula 3.11 to obtain m_k. Secondly, the system calculates the distance to the new point by multiplying distance and velocity, hence, obtaining the circumference of all the points that are at that distance from the actual point. The segment intersects the circumference in two points, being one of them the required point. Finally, the sign of the velocity feature is used to choose between the points. Figure 3.8 shows the process used to

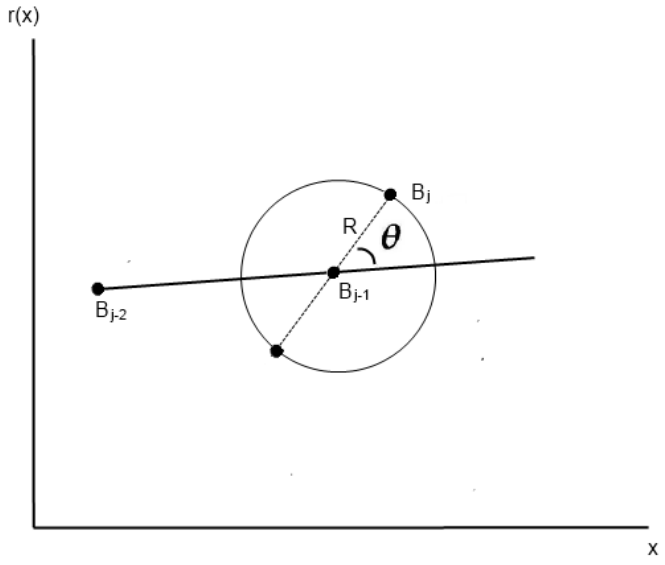

Figure 3.8: Process to rebuild β_j

rebuild the trajectory. R represent the radius of the circumference, obtained by multiplying velocity and duration.

3.4.2 Summary

In this section, three different features are obtained for every breakpoint in each trajectory. The representation build using these features is invariant to rotation and orientation and can be used to perform a wide variety of analysis. Moreover, it is possible to rebuild the original trajectory using the features proposed.

4

Experiments and Results

The objective of this chapter is to present the experiments done to adjust and test the developed system. Moreover, the chapter introduces the databases used to evaluate the stages explained in the previous chapter. The conclusions extracted from the results of the run experiments are commented in chapter 5. There is an evaluation point for each stage described in chapter 3. Additionally a final evaluation is included to test the performance of the system used to discriminate between cars and pedestrians. The chapter is divided as follows:

1. **Dataset:** brief introduction to the datasets used to evaluate the system. The ground truth used in the experiments is also presented.

2. **Preprocessing module:** description of the experiments designed to check the effectiveness of each stage of the preprocessing module.

3. **Segmentation module:** presents the experiment run to set the thresholds related to the detection of breakpoints. Moreover, the efficacy of the filtering stage decreasing the number of breakpoints necessary to represent the trajectory is also evaluated.

4. **Trajectory features:** tests the capacity of the implemented features to represent the original trajectory from which they were extracted. The accuracy in the reconstruction of the original trajectory is also checked.

5. **Trajectory representation:** the tests consist in evaluate the performance of the proposed system for object recognition.

6. **Real-time performance:** test the time performance of the system to evaluate the possibility of working in real-time applications.

4.1 Dataset

In order to consider several scenarios, the system has been tested with surveillance videos from two different datasets. The videos included, present a variety of situations and conditions including indoors/outdoors scenarios and parked/road environments.

Figure 4.1: Examples from i-LIDS dataset

Figure 4.2: Examples from TRECVID 2008 dataset

- i-LIDS dataset: i-LIDS consists of a library of CCTV video datasets with different six scenarios including parked vehicle detection and road surveillance scenarios. All the videos are recorded in outdoors environments. The footage accurately represents real operating conditions. The Figure 4.1 shows some examples of i-LIDS videos.
- TREVCID 2008 dataset: TREC Video Retrieval Evaluation (TRECVid) is an international benchmarking activity to promote research in video information retrieval by providing a large test collection and uniform scoring procedures. The 2008 dataset includes several indoors scenarios, most of them in crowded environments.

4.1.1 Ground truth

The ground truth has been built by selecting a small sized set extracted from the dataset and manually annotating it for two different concepts: car and person. In total, 171 objects were included in the ground truth dataset. The dataset includes most of the scenarios available in the datasets described above. Furthermore, the training set used in the experiments described in 4.5 is obtained from the ground dataset. The ground truth was built using the Viper-GT tool [25]. This tool generates an output file with information of the manually annotated objects including position, size and frame of every point. To extract and reformat this information in order to compare it with the one extracted from the system was necessary to implement a new program. The platform selected to implement it was Java. Finally, the data obtained was used

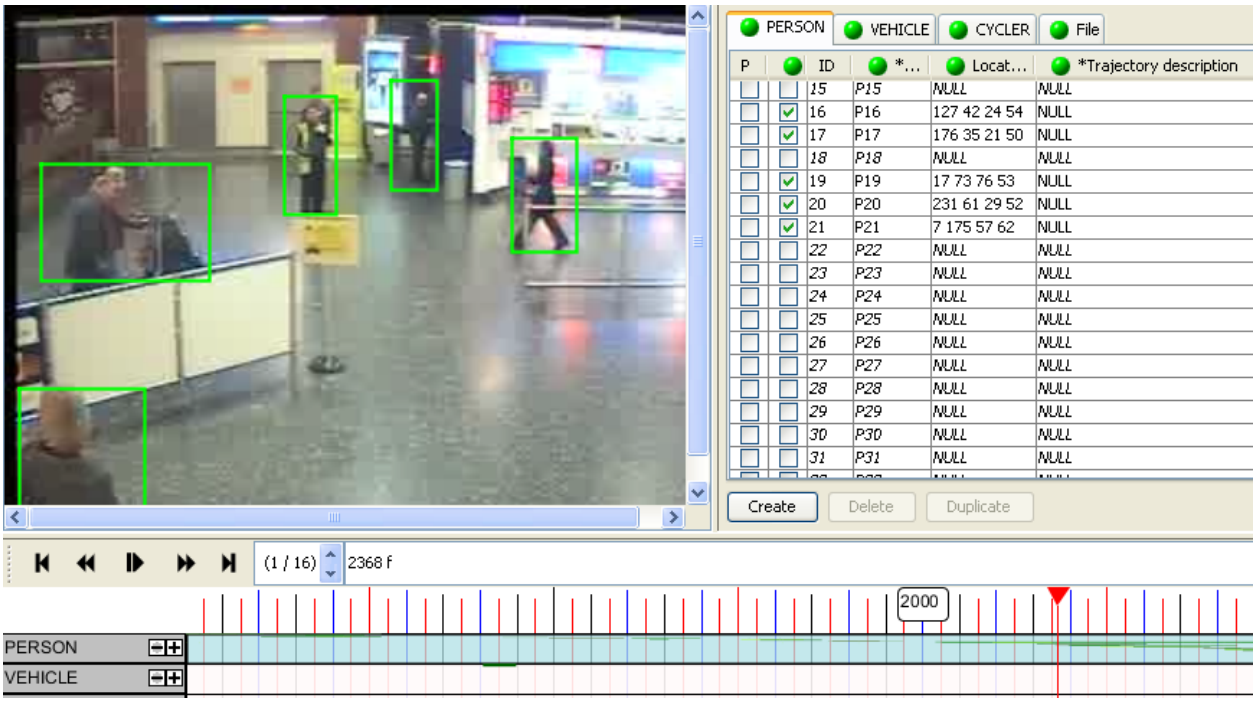

Figure 4.3: Annotating a video using Viper-GT tool

in several of the experiments explained in the next paragraphs.

4.2 Preprocessing step

4.2.1 Separation of different trajectories parameters

When paths from different objects cross or pass near each other, sometimes are confused as belonging to the same object. To detect the trajectories joined this way, the system analyses each trajectory and checks if any of the conditions defined in section 3.2.1 is satisfied. The conditions check the Euclidean distance between every two consecutive points (P_k, P_{k+1}), $\epsilon_{k,k+1}$, and the β angle defined in section 3.3.2. If $\epsilon_{n,n+1} > \epsilon$ or $\beta_k > \beta$, the trajectory is divided in two from P_k.
The figure 4.4 shows, in the part above, a situation where the paths of two different cars pass near each other and are confused to belong to the same trajectory. The figures below show the trajectory extracted by the motion analysis component, and the trajectories after the Separation of Different Trajectories step.
To evaluate this step, the number of joined trajectories after the Motion Component Analysis was counted for the test videos. The parameters were evaluated in terms of how many of these cases were able to solve. Additionally, the results were checked to find incorrect separations. The measures used to evaluate the effectiveness of this step are defined below:

- Hit rate:

$$\text{Hit rate} = \frac{\text{Resolved cases}}{\text{Total bad joined cases}} \tag{4.1}$$

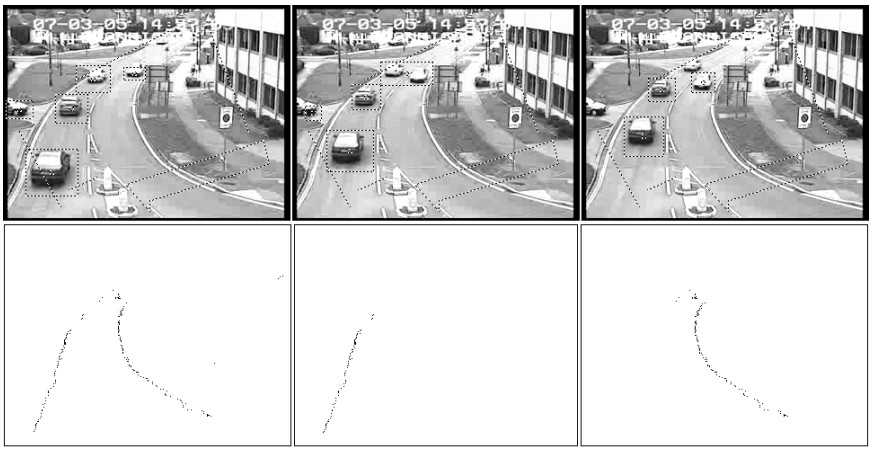

Figure 4.4: Example of trajectory separation

Hit rate	Error rate
0.916	0

Table 4.1: Hit and error rates for Separation of different trajectories experiments

- Error rate:

$$\text{Error rate} = \frac{\text{Separated paths belonging to the same object}}{\text{Total number of trajectories}} \tag{4.2}$$

To set the value of ϵ is important to take into account two important facts: the first, is the possibility of find occlusions in the trajectory; the second, that the higher the distance between two points, the less the probability of belonging both to the same trajectory. Thus, a low value for ϵ will result in the separation of most of the trajectories with little occlusions, while a high value will make the complete stage useless. Regarding these two points, ϵ was varied in the experiments to a range of values and was finally set to the value with the best performance in terms of solved cases. The election of a value for the threshold β is based on the idea that the changes in the paths of cars and pedestrians are normally smooth, and the sharp turns are rare in the surveillance videos environments. The experiments were run with an initial value for β and using increments of 10 degrees to the final value.
The table 4.2.1 shows the results obtained from the analysis of the videos included in the ground truth. The proposed solution was able to resolve more than 90% of the cases. Moreover, the solution helps to eliminate noise from the trajectories in certain circumstances as shown in figure 4.5.

4.2.2 Occlusion conditions

Objects trajectories are split when the detected objects in the videos get occluded by either inanimate or detected objects. In order to recover the original trajectory and prevent the effects of the occlusions, a set of conditions is defined. These conditions are presented and explained in detail in section 3.2.2. The figure 4.6 shows a pedestrian being occluded by a cyclist. Below,

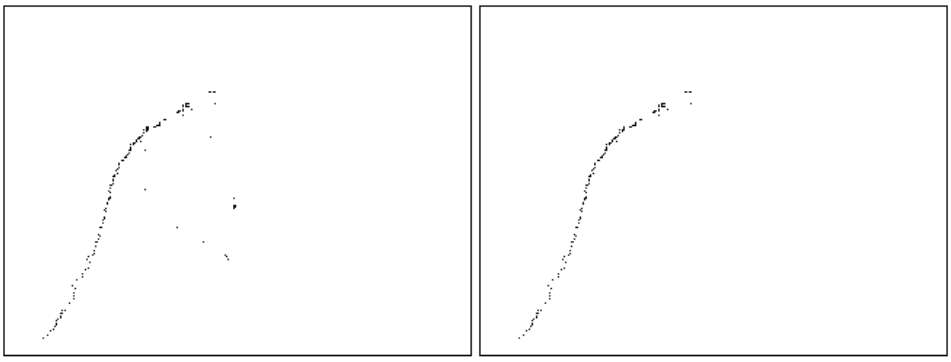

Figure 4.5: Example of noise removal in Separation of Different Trajectories step

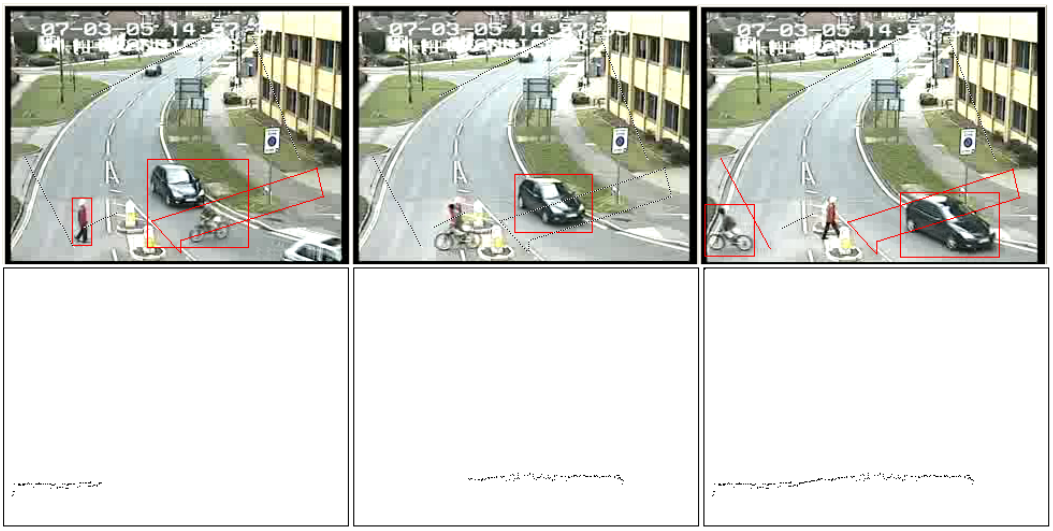

Figure 4.6: Example of occlusion and trajectory joining

the paths detected by the motion analysis component and the final joined path are represented. The measures used in this experiments are similar to the ones used in the previous section. The difference is that the occlusions were count directly on the annotations. The effects of the variables values are discussed in the next paragraphs:

- Spatio-temporal proximity: to join two paths, first is necessary to check their spatio-temporal proximity. Two thresholds N and F were implemented to check this condition. A small value for N decreases overly the chance to make joints, due to the merge of objects in the motion analysis component. This happens when an object occults other, and the motion analysis component treat them like a new object for an interval of time δt. If δt is bigger enough, the chance of recover the trajectory by joining the separated paths is very low. Therefore, a big value ensures to resolve more occlusions,in spite of increasing the chance to make an incorrect joining as well. In the case of F, the reference chosen was a maximum of 1 second between paths.

- Similar direction: this condition is necessary to avoid joining close paths of similar objects that move in different directions. To set the value for the threshold α_2, several values were tested in the experiments. Finally the value was set trying to minimize the error rate.

- Similar size: the Spatio-temporal proximity condition ensures that the paths being analysed must be close in distance. This means the effect of the image depth can not affect much to the size of the object. Thus, if the two paths belong to the same object, their size must be similar. To ensure the paths have a similar average size a maximum size increment threshold is implemented. The size increment measurement ΔS is defined in formula 5.1:

$$\frac{|S_1 - S_2|}{S_1} < \Delta S \qquad (5.1)$$

  In the experiments, to find an appropriate value for ΔS; examples of the most common objects, pedestrians and vehicles, were extracted in positions separated N (see the first condition spatio-temporal proximity). For each sample, a size increment was calculated. Then, the value of ΔS was set using the results shown in table 4.2.2.

	Average size increment ΔS
Pedestrians	0.013
Vehicles	0.035

Table 4.2: Average ΔS for cars and pedestrians

Hit rate	Error rate
0.125	0.008

Table 4.3: Occlusion scores

The table 4.2.2 shows the results for occlusion handling based on the analysis of the selected video dataset. The proposed solution is effective in only 12.5% of the cases.

4.2.3 False objects removal conditions

In section 3.2.4, the steps to perform the false object removal are explained in detail. Some of the conditions that the objects need to check to be considered proper objects are defined as thresholds or variables. The performance measures used in this experiments are again the ones defined in Section 4.2.1. In the next paragraphs, the values for these thresholds and variables are set and justified.

- Minimum stroke: a measurement based in Euclidean distance is used to calculate the stroke. The reason beneath the implementation is to filter most of the false objects caused by light reflections, which are frequently static but active. Low values increase the number of false objects in the system while a bigger value can discard real trajectories as if they were false.
- Minimum size: the minimum size threshold was established using examples of pedestrians in the farthest distance from the camera as possible. The experiments indicated that is one the less important factors in the task of eliminating false objects.

Hit rate	Error rate
0.875	0.041

Table 4.4: False object removal scores

The table 4.2.3 shows the results for false object removal. The implemented solution is effective in the 87.5% of the cases.

4.3 Angle-based trajectory division step

4.3.1 Beta and gamma thresholds selection

To adjust the thresholds to the values best fitted to the system, the trajectories were separated into three groups: long (>120 points), medium (>60 and <120 points) and short trajectories (<60 points). For each group, 10 trajectories were selected from videos belonging to the test dataset, including both trajectories with big deviations and plain trajectories. In the experiments, both the beta and gamma threshold were set to different values. Beta low values resulted in bad approximations to the trajectories, while values over a certain angle do not have significance effects in the number of breakpoints.
For each pair of values, two output parameters were analysed: a similarity score between the rebuilt trajectory and the preprocessed one, based in Euclidean distance; and the number of breakpoints used to represent the trajectory. The objective was to find a good relation between the two parameters, leading to a compact and accurate representation. To compare the preprocessed trajectory with the one rebuilt from the features, the values between each reconstructed breakpoint were calculated using linear interpolation. The formula used to calculate the similarity score is detailed below:

$$\frac{\sum_{i=1}^{i=n} \sqrt{\left(x_i^a - x_i^b\right)^2 + \left(y_i^a - y_i^b\right)^2}}{n} \tag{4.3}$$

where (x_i^a, y_i^a) are the *ith* coordinates of the original trajectory, (x_i^b, y_i^b) the *ith* coordinates of

the rebuilt trajectory and n is the number of points of that particular trajectory.
As shown in Figure 4.7, the result of using a beta threshold of 50 degrees is a reduction in the number of breakpoints used to characterize the trajectory, although implies an increase in the distance measurement. This increase is only relevant in the medium trajectories, however, the score value for $\beta = 50$ is close to the values for short trajectories. In the case of the long trajectories, values between $\gamma = 40$ and $\gamma = 50$ are even lower for $\beta = 50$ than for $\beta = 30$. Therefore, the selected value was $\beta = 50$.
Gamma value affects in a similar way each group of trajectories, increasing this variable always leads to decrease the number of breakpoints and increase the score value. Nonetheless, as shown in figure 4.8, from $\gamma = 40 - 50$ the reduction in the number of breakpoints is small compared to the big increase in the score of the long trajectories (figure 4.7). The average score for the long trajectories doubles its value between $\gamma = 40$ and $\gamma = 80$, only decreasing in 2 the average number of breakpoints. For the rest of the cases (short and medium trajectories), none of the parameters changes substantially between $\gamma = 40$ and $\gamma = 60$. Thus, the final value was set to $\gamma = 45$.
The table 4.3.1 shows the value of the output parameters for $\beta = 50$ and $\gamma = 45$

	Average score	Average number of breakpoints
Long trajectories	3.79	9.8
Medium trajectories	3.24	6.6
Short trajectories	2.77	3.9

Table 4.5: Output parameters for $\beta = 50$ and $\gamma = 45$

4.3.2 Filter optimization

In the stage of trajectory smoothing, the result of the filtering depends on two parameters: the polynomial degree (P) and the filter length (N) (see section 3.2.3). Starting from the values recommended in [12], $P = 4$ and $N = 21$ a number of experiments were performed to test the effectiveness of the filtering and try to find best parameters values than the initial ones. New possible values for these parameters are conditioned to the restrictions presented in section 3.2.3:

- N must be odd.
- N must be smaller than the number of points of the trajectory to be filtered.
- $N > P$.

These conditions restrict the possible values of N, to the odd values between P and the value set in the False objects removal section, which is the minimum point value to consider the object valid (see section 4.2.3).
The first experiment tries to find out if the application of filtering is really decreasing the number of breakpoints necessary to represent the trajectory. All the videos contained in the test dataset were analysed twice: one including filtering and another time without the filter algorithm. The results are contained in the table 4.3.2. The objective of the second experiment was to improve the results obtained in the first experiment changing the filter parameters. To analyse the behaviour of the filter depending on N, this parameter was set to different values. The polynomial degree was maintained to the initial value. The best results occurred for the highest values of N. The second part of the experiment, consisted in finding the best P value

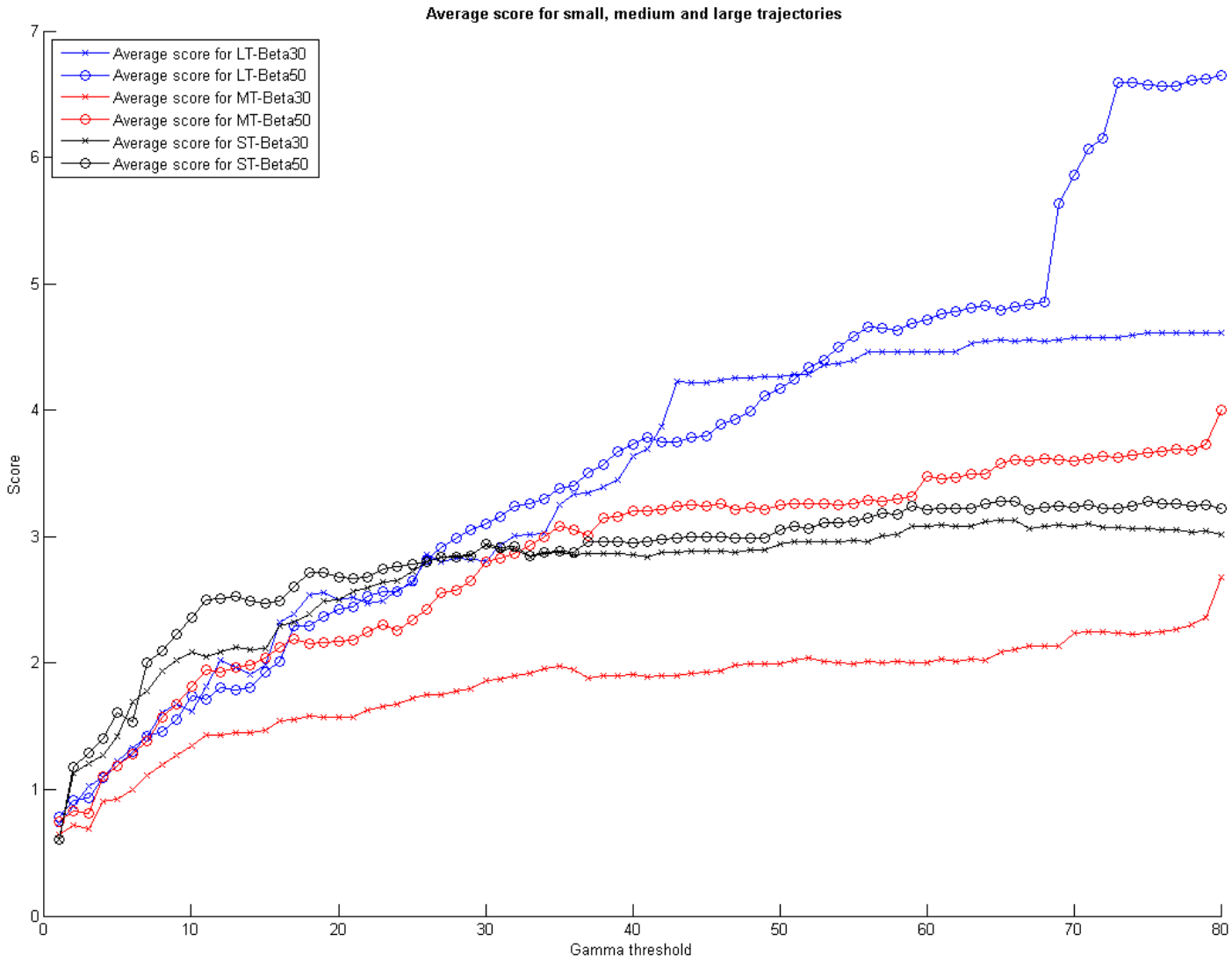

Figure 4.7: Average score for $\beta = 30$ and $\beta = 50$

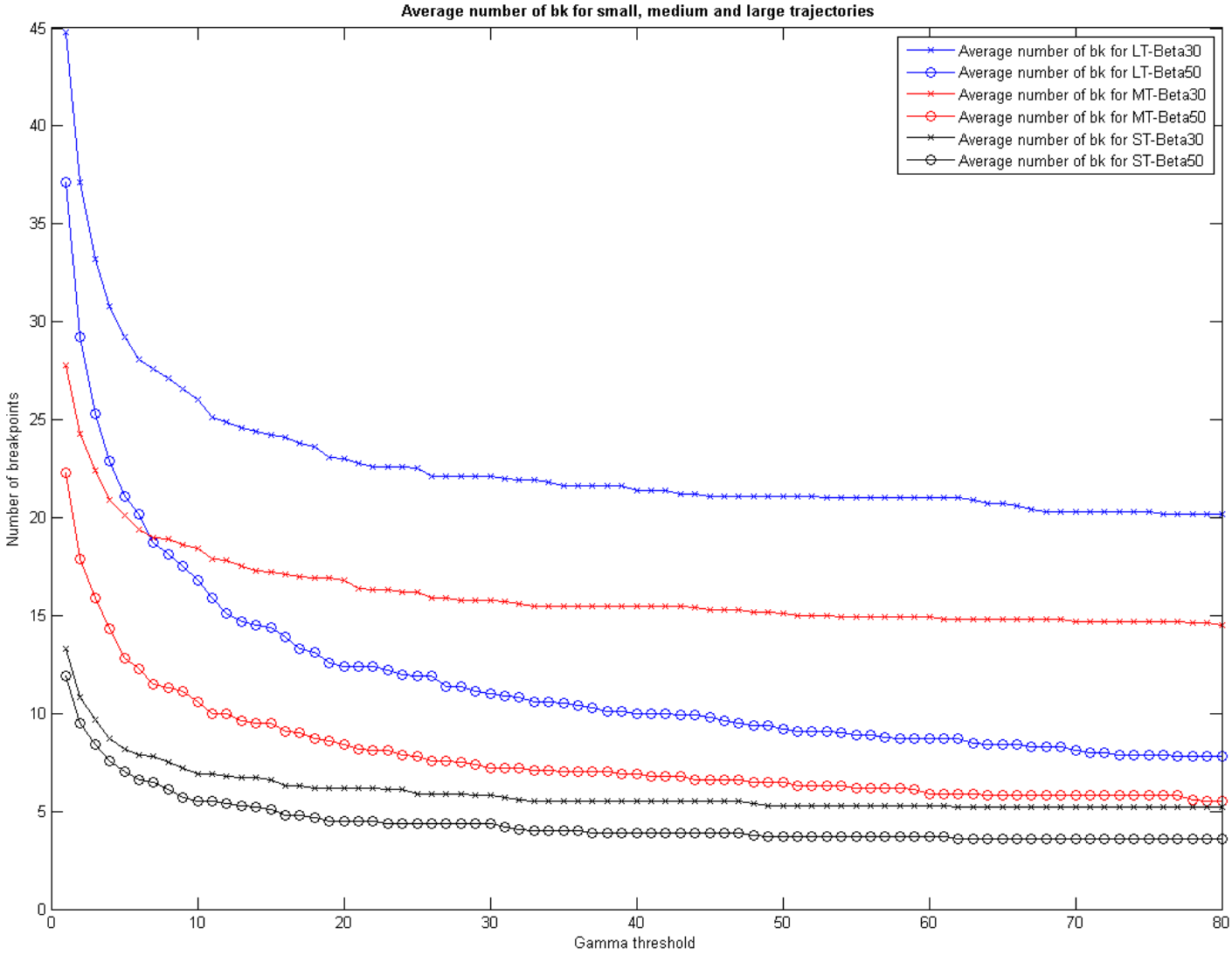

Figure 4.8: Average number of breakpoints for $\beta = 30$ and $\beta = 50$

Breakpoints with filtering	Breakpoints without filtering
8.27	15.38

Table 4.6: Number of breakpoints using or not filtering

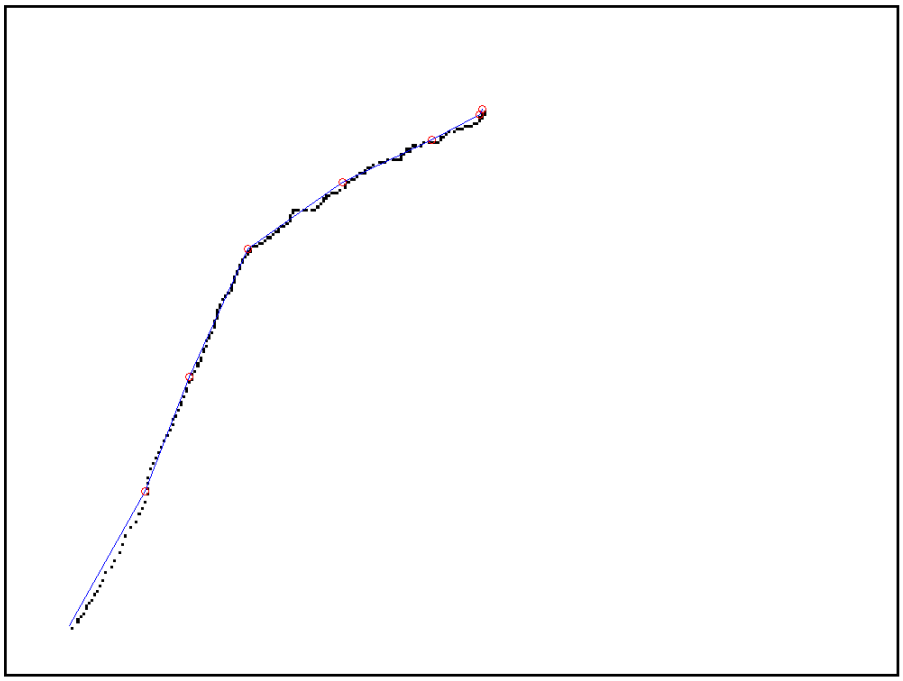

Figure 4.9: An example of a rebuilt trajectory

for the selected N. The final values were tested against the initial values. The results showed that the initial values perform better.

4.4 Trajectory rebuilding

The objective of this experiment is to determine how accurate are the proposed features representing the information contained in the trajectories. As stated in section 3.4, three different features are extracted: angle, velocity and duration. To check if these features can approximate the original trajectory, an algorithm has been implemented. The algorithm starts from the breakpoints, and using the features related to them, is able to reconstruct the original trajectory. In most of the cases, the reconstructed trajectory is almost identical to the original. However, in some cases, the reconstructed one is not that similar, but it conserves the main characteristics such as global direction and directionality. The error between the trajectories can be measured using the process described in Section 4.3.1 for the score measure. In the figure 4.9 there is a representation of both the filtered trajectory (in black) and the rebuilt trajectory (in blue). The red points represent the breakpoints.

4.5 Object recognition based on velocity

This experiment pursues to determine if the velocity feature extracted is effective to discriminate between vehicles and pedestrians. The main reason to select this feature is that is reasonably think that there will be big differences in the velocity of cars and pedestrians even in an urban environments. Using the examples hand-annotated (from videos of i-LIDS dataset) for the ground truth, a number of velocity vectors is used to train a system of SVMs [26]. The test dataset is compose of extracted objects from the i-LIDS dataset.The reason for choosing all the test trajectories from the same dataset is that the velocity is extracted in pixels/second which means that is high dependent on the distance between the camera and the object.
In the measurements extracted from the SVM system, the difference expected between the velocity of vehicles and pedestrians was not enough to make a correct classification. The classifier was unable to distinguish between the two concepts of object, thus demonstrating the inefficacy of velocity to perform object recognition.

4.6 Real-time performance

The Motion Analysis Component described in chapter 3 can be used to work in real-time applications with low resolutions. The objective of this experiment is evaluate if the whole system is able to work in real-time. To measure this capacity, the system was run with different videos, estimating the performance time of the implemented parts of the system over the performance time of the Motion Analysis Component. The results show that the total time of performance is an $8 - 10\%$ more than the Motion Analysis Component performance time. This result is promising, despite of not been sustained using a real-time application.

5

Conclusions and future work

5.1 Conclusions

Video surveillance related applications have emerged as a technology covering the increasingly need for automated security systems. In these applications, trajectory-based techniques are widely used to model the behaviour of moving objects in a big variety of scenarios. The objective of this thesis was to design and develop a robust system capable of obtain an accurate representation for these trajectories. Moreover, the implemented system had to be able to reconstruct the original trajectory with the less error possible.
The results of the experiments probe that the implemented system works good in most of the scenarios used to test the performance. The system is robust against most of the problems encountered in video surveillance applications, despite of the results for the occlusion case, which are quite improvable. The path division technique has demonstrate to be effective in finding the points which held the information needed to describe the whole trajectory. This fact, together with the good performance of the selected features in the trajectory reconstruction step assures an accurate reconstruction of the original trajectories. However, the velocity feature demonstrated to be ineffective for object recognition.
The results ensure, therefore, that the implemented system meets the objectives presented in Chapter 1. Moreover, the experiments show promising results for real-time performance which is an important requirement for many actual systems. The conclusions for each of the experiments for each step are presented below:

- Preprocessing step: the experiments were ran to test the performance of the system against the most common challenges in video surveillance systems. The experiments probe that the implemented systems deals good in most of the cases, with results near to a 90% of solved cases, with the exception of the occlusion handling solution. The geometrical and temporal conditions implemented to solve occlusions are not effective in most of the cases. The addition of a data smoothing step has been proof useful to decrease the number of breakpoints necessary to represent the trajectory.
- Angle-based division step: the division of the trajectories into subpaths reduced the number of points necessary to represent the trajectory to less a 10% of the original ones. Also, this division in subpaths makes the system more flexible, allowing to work with the subpaths or the entire reconstructed trajectories.

- Trajectory rebuilding: the experiments demonstrate that the reconstructed trajectories can approximate the original ones with little error, which at the same time is a proof of the good performance of the extracted trajectories in trajectory rebuilding. In addition, the amount of information to represent the trajectories has been reduced, thus meeting the objective of finding an accurate and simple representation.
- Object recognition based on velocity: the results for the test of effectiveness of the velocity to discriminate between vehicles and pedestrians were poor, thus, an indication that velocity has no discrimination power. There was no real difference between the velocity of cars and pedestrians, which can be explained due to the low speed of the vehicles in urban areas. Additionally, most of the videos were filmed in places such as pedestrian crossings and traffic lights, contributing to slow the average speed of the vehicles.

5.2 Future work

Based on the work done in this thesis, two options are open to continue: enhance the system and/or extend it developing a high-level application over it. The principal points to enhance in the system include: robustness against occlusions, extension of the ground truth and increase the number of features of the system.
In the case of occlusion handling, the proposed solutions have not been effective. As most of the actual work in occlusion handling is based on robust tracking algorithms, two options must be considered: change the current tracking algorithm for other more robust against occlusions or find a way to improve the implemented one. Change the algorithm can result in worse results depending on the substitute but the complexity of the current algorithm makes difficult to improve the actual implementation.
The features used in the system have probed their effectiveness, but most of the actual high level applications need to extract more information from the videos to perform correctly. Thus, the implementation of new features can facilitate further work on high level concepts.
The extension of the ground truth can be an important enhancement. Increasing object examples can lead to more accurate and complete experiments in the future.
The development of a high level application over the system must be studied carefully, because depending on the election the system will need to be changed in consequence. Some possible applications to extend the presented approach include event detection, crowd analysis or object identification.

Bibliography

[1] Omar Javed and Mubarak Shah. Tracking and object classification for automated surveillance. *ECCV '02 Proceedings of the 7th European Conference on Computer Vision*, 4:343–357, 2002.

[2] Nicola Piotto, Nicola Conci, and Francesco G. B. De Natale. Syntactic matching of trajectories for ambient intelligence applications. *IEEE Transactions on Multimedia*, 11:1266–1275, 2009.

[3] Gill M., Allen A., Jessiman J., Swain D., Hemming M., Kara D, and Little R. Methods in assesing the impact of cctv. *Home Office Report*, 17, 2005.

[4] M. H. Sedky, M. Moniri, and C.C. Chibelushi. Classification of smart video surveillance systems for commercial applications. *IEEE Conference on Advanced Video Signal Based Surveillance*, 1:638–643, 2005.

[5] H. U. Keval and M.A. Sasse. Man or gorilla? performance issues with cctv technology in security control rooms. *16th World Congress on Ergonomics Conference, International Ergonomics Asociation*, pages 10–14, 2006.

[6] M. Jerian, S. Paolino, F. Cervelli, S. Carrato, A. Mattei, and L. Garofano. A forensic image processing environment for investigation of surveillance video. *Forensic Science International*, pages 167–170, 2007.

[7] Stefan Muller-Scheneiders, Thomas Jager, Harmut S. Loos, Wolfgang Niem, and Robert Bosch. Performance evaluation of a real-time video surveillance system. *2nd Joint IEEE International Workshop on Visual Surveillance Evaluation of Tracking and Surveillance*, 1:137–143, 2005.

[8] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. *ICCV*, page 255, 1999.

[9] Sen-Ching, S. Cheung, and Chandrika Kamath. Robust techniques for background subtraction in urban traffic video. *Visual Communications and Image Processing*, 5308:881–892, 2004.

[10] A. Prati, I. Mikic, M. Trivedi, and R. Cucchiara. Detecting moving shadows: algorithms and evaluation. *IEEE Transactions on Pat*, 25:918–923, 2003.

[11] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Computing Surveys*, 38, 2006.

[12] Ivan Ivanov, Frederic Dufaux, Thien M Ha, and Touradj Ebrahimi. Towards generic detection of unusual events in video surveillance. *AVSS '09 IEEE Conf. on Advanced Video and Signal Based Surveillance*, 6:61–66, 2009.

[13] F. Fleuret. Fixed point probability field for complex occlusion handling. *Proceedings of the Tenth IEEE International Conference on Computer Vision*, 1:694–700, 2005.

[14] R. Cucchiara, C. Grana, and G. Tardini. Track-based and object-based occlusion for peple tracking refinement in indoor surveillance. *VSSN '04 Proceedings of the ACM 2nd international workshop on Video surveillance & sensor network*, pages 81–87, 2004.

[15] Hannah M. Dee and Sergio A. Velastin. How close are we to solving the problem of automated visual surveillance systems? *Machine vision and applications*, 19:329–343, 2008.

[16] C. Stauffer. Estimating tracking sources and sinks. *Computer vision and pattern recognition*, 4:35, 2003.

[17] Brendan T. Morris and Mohan M. Trivedi. A survey of vision-based trajectory learning and analysis for surveillance. *IEEE Transactions on circuits and systems for video technology*, 18:1114–1127, 2008.

[18] Jianwen Luo, Kui Ying, Ping He, and Jing Bai. Properties of savitzky-golay digital differentiator. *Digital Signal Processing*, 15:122–136, 2005.

[19] F. Bashir, W. Qu, A. Khokhar, and D. Schonfeld. Hmm-based motion recognition system using segmented pca. *Proceedings IEEE in Computer Visual Pattern Recognition*, 3:1288–1291, 2005.

[20] Fatih Porikli and Tetsuji Haga. Event detection by eigenvector decomposition using object and frame features. *Conference on Computer Vision and Pattern Recognition Workshop*, 7:114, 2004.

[21] C. Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:747–757, 2000.

[22] C. Flanagan. The bresenham line-drawing algorithm, October 1996. http://www.cs.helsinki.fi/group/goa/mallinus/lines/bresenh.html.

[23] Gilbert Strang. *Linear Algebra and Its Applications*. Brooks Cole, 1988.

[24] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[25] Language and University of Maryland Media Processing Laboratory. A video processing performance evaluation tool: Viper-pe. viper-toolkit.sourceforge.net/products/pe/, 2003.

[26] T. Joachims. Svmlight support vector machine, August 2008. svmlight.joachims.org.

A

Presupuesto

1)	**Ejecución Material**	
	• Compra de ordenador personal (Software incluido)	2.000 €
	• Material de oficina	250 €
	• Total de ejecución material	2.250 €
2)	**Gastos generales**	
	• sobre Ejecución Material	360 €
3)	**Beneficio Industrial**	
	• sobre Ejecución Material	135 €
4)	**Honorarios Proyecto**	
	• 1800 horas a 15 €/ hora	27000 €
5)	**Material fungible**	
	• Gastos de impresión	200 €
	• Encuadernación	160 €
6)	**Subtotal del presupuesto**	
	• Subtotal Presupuesto	30.105 €
7)	**I.V.A. aplicable**	
	• 21% Subtotal Presupuesto	6322.05 €
8)	**Total presupuesto**	
	• Total Presupuesto	36427.05 €

Madrid, Enero 2013

El Ingeniero Jefe de Proyecto

Fdo.: Gonzalo Varela Bartrina

Ingeniero Superior de Telecomunicación

B

Pliego de condiciones

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un *Accurate trajectory representation for video surveillance*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales.

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares.

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.

C

Introducción

C.1 Motivación

En los últimos años, se ha incrementado sensiblemente la presencia de sistemas de vídeo vigilancia en muchas partes del mundo. El incremento en la demanda de seguridad y la innovación tecnológica han posibilitado la expansión de estos sistemas. Hoy en día se pueden encontrar cámaras tanto en espacios públicos como aeropuertos o estaciones, como en espacios privados tales como áreas industriales, tiendas o en los hogares. Esto supone que, cada día, se graba una enorme cantidad de información.

La mayoría de los sistemas implementados son monitorizados en su mayor parte por personal de seguridad, lo que conlleva una pérdida de efectividad y una serie de costes que se explicarán a continuación. En primer lugar, la enorme cantidad de material filmado implica que es imposible mantener un control efectivo sobre el material filmado sin invertir una cantidad muy grande de dinero en recursos humanos. En segundo lugar, la monitorización no-automática de los contenidos de los vídeos significa que será necesaria una segunda revisión de los mismos en el caso de querer extraer nueva información del vídeo. Ante estos problemas, los sistemas y aplicaciones de vídeo automáticos están surgiendo como una solución para mejorar la calidad y las oportunidades futuras de los sistemas de vídeo vigilancia.

Sin embargo, hay una serie de problemas comunes a todos los sistemas de vídeo vigilancia con los que es necesario tratar. Estos problemas incluyen: cambios de iluminación, presencia de ruido en la grabación o problemas asociados a la baja calidad de los vídeos filmados. Adicionalmente, pueden aparecer otros problemas relacionados con el contenido del vídeo en cuestión como oclusiones u objetos inanimados movidos por el viento que pueden empeorar el rendimiento de los sistemas asociados.

Muchas de las aplicaciones propuestas o implementadas en el área de la vídeo vigilancia se basan en la información extraída a partir de la trayectorias de los objetos que se mueven en los vídeos analizados. Esta información extraída, de naturaleza espacial y temporal, es luego utilizada por aplicaciones de alto nivel para realizar tareas como monitorización del tráfico, detección de eventos inusuales o discriminación de objetos detectados. Dada la dependencia de estos sistemas a la información contenida en las trayectorias, es importante lograr una extracción y representación precisa de las mismas. Por tanto, cualquier mejora en la extracción y representación de las trayectorias es paso hacia delante en la mejora de la calidad y los resultados de muchas aplicaciones en el campo de la vídeo vigilancia.

C.2 Objetivos

El objetivo de este proyecto es el diseño e implementación de un sistema capaz de extraer y representar de forma precisa las trayectorias de los objetos en movimiento de los vídeos analizados. Además, se pretende que la implementación sea capaz de afrontar los desafíos comunes a todo sistema automático de vídeo vigilancia como son el tratamiento de ruido, oclusiones y otros problemas relacionados. Para la consecución de estos objetivos, es necesario completar una serie de objetivos parciales que se muestran a continuación:

- Estudiar en la literatura la viabilidad de las soluciones actuales a los problemas de ruido, oclusión, reducción de la dimensionalidad y otros problemas asociados a los sistemas basados en la extracción de trayectorias.
- Implementar un método para la detección de cambios espaciales y temporales, haciendo posible la fragmentación de la trayectoria en partes con un comportamiento homogéneo internamente, facilitando de esta manera la posterior representación de las trayectorias.
- Encontrar e implementar una serie de características que representen las trayectorias de los objetos detectados, centrándose en la precisión y complejidad de las mismas.
- Evaluar el funcionamiento y comportamiento de cada una de las técnicas implementadas.
- Evaluar el funcionamiento del sistema completo usando bases de datos de vídeo vigilancia estándar, a régimen de poder dotar de significado a los resultados.

C.3 Estructura del documento

El documento se ha estructurado como se expone a continuación:

- Capítulo segundo: en este capítulo se realiza el estudio y la exposición de los problemas y desafíos presentes en los sistemas de vídeo vigilancia. Además se exponen las principales técnicas encontradas en la literatura para el análisis de los objetos en movimiento en vídeos, solución de los problemas de oclusión , mejora de la calidad de las trayectorias y representación de éstas últimas.
- Capítulo tercero: en este capítulo se introduce el sistema propuesto, presentando cada etapa del mismo desde el análisis de movimiento hasta la etapa de evaluación. Los algoritmos y técnicas son explicados a fondo, detallando las ventajas y desventajas de la implementación elegida. Adicionalmente, se analizan los problemas y limitaciones hallados en su implementación, explicando para cada caso la solución adoptada.
- Capítulo cuarto: en este capítulo se presentan los experimentos llevados a cabo para evaluar el sistema. Las bases de datos elegidas para realizar los experimentos son presentadas al inicio del capítulo.
- Capítulo quinto: las conclusiones extraídas y las posibilidades de trabajo futuro se analizan en este capítulo.

D

Conclusiones y trabajo futuro

D.1 Conclusiones

Las aplicaciones y sistemas de vídeo vigilancia han surgido para cubrir la demanda de sistemas de vídeo seguridad automáticos. Entre estas aplicaciones, una importante proporción utiliza técnicas basadas en la extracción de trayectorias para analizar y modelar el comportamiento de los objetos en movimiento en una gran variedad de escenarios. El objetivo final de este proyecto era diseñar y desarrollar un sistema robusto capaz de generar una representación precisa de las trayectorias. Así mismo, el sistema debía ser capaz de reconstruir las trayectorias originales con el menor error posible. Los resultados de los experimentos muestran que el sistema propuesto cumple los objetivos en la mayor parte de los escenarios usados para probar su rendimiento. El sistema es robusto contra casi todos los problemas encontrados en vídeo vigilancia, exceptuando el caso de las oclusiones, donde el rendimiento es bastante bajo.
La técnica de división de trayectorias ha mostrado ser efectiva para encontrar los puntos clave de donde se puede extraer la información para describir el comportamiento de la trayectoria. Este hecho, junto al éxito de la reconstrucción partiendo de las características extraídas, hace posible la reconstrucción de las trayectorias originales. Sin embargo, la característica velocidad demostró ser completamente ineficaz a la hora de realizar una tarea de reconocimiento de objetos.
Los resultados aseguran, por tanto, que el sistema presentado cumple la mayoría de los objetivos marcados presentados en el capítulo 1. Adicionalmente, los experimentos sobre el tiempo de ejecución adelantan un buen rendimiento en sistemas de tiempo real de baja resolución (donde el Motion Analysis Component puede trabajar en tiempo real).
Las conclusiones parciales para cada uno de los experimentos se muestran a continuación:

- Experimentos de la etapa de pre-procesamiento: los experimentos llevados a cabo dieron buenos resultados contra los problemas más comunes en vídeo vigilancia, con resultados cercanos al 90% de casos resueltos. La excepción es el caso de la oclusión, donde los resultados fueron bastante peores. Las condiciones geométricas y temporales implementadas para detectar los casos de oclusiones resultados ineficaces en la mayoría de los casos. Por el contrario, el suavizado por filtrado de las trayectorias demostró ser muy eficaz disminuyendo el número de puntos de corte necesarios para representar la trayectoria.

- Experimentos de la etapa de división de la trayectorias: esta etapa permitió representar (reconstruyendo posteriormente) las trayectorias con tan solo el 10% aproximadamente de los puntos originales. Además, la división de las trayectorias permite trabajar con las trayectorias reconstruidas o con las partes surgidas de la división.
- Experimentos de la etapa de reconstrucción de las trayectorias: los experimentos muestran que es posible reconstruir las trayectorias originales, manteniendo las características globales y aproximándose bastante al recorrido original. Así mismo, la cantidad de información necesaria para representar las trayectorias se ha reducido enormemente, cumpliendo los objetivos marcados en el capítulo 1.
- Experimento de reconocimiento de objetos basado en la velocidad: el experimento llevado a cabo con el fin de determinar si la característica de velocidad era útil para realizar tareas de reconocimiento de objetos arrojó pobres resultados. El clasificador no fue capaz de discriminar entre velocidades de peatones y vehículos. Este hecho puede ser explicado en parte debido a la baja velocidad de los vehículos en entornos urbanos, donde las cámaras frecuentemente estaban posicionadas cerca de semáforos y pasos de peatones.

D.2 Trabajo futuro

Una vez extraídas las conclusiones finales, se abren dos opciones: mejorar el sistema y/o extenderlo desarrollando una aplicación de alto nivel sobre el mismo. Los principales puntos a mejorar son: la robustez del sistema contra oclusiones, incrementar el número de características extraídas y la ampliación de los bancos de pruebas, tanto el *ground truth* como las bases de datos utilizadas.
En el caso de la oclusión, la solución propuesta ha demostrado ser ineficaz. Dado que la mayoría del trabajo frente a la oclusión se basa en el diseño de algoritmos de *tracking* robustos, hay dos decisiones posibles. La primera consistiría en cambiar el algoritmo actual por otro más robusto, mientas que la segunda supone intentar mejorar el algoritmo implementado. Cambiar el algoritmo requeriría probablemente reajustar partes del sistema pero la alta complejidad del algoritmo actual hace que su mejora pueda resultar una tarea bastante complicada.
Las características elegidas para la implementación han probado su utilidad, sobre todo en cuanto a representación de trayectorias. Sin embargo, muchas de las aplicaciones de alto nivel requieren más información de la que estas características pueden proporcionar, por lo que la implementación de nuevas características puede facilitar mucho el trabajo posterior.
La extensión de las bases de datos de pruebas y sobre todo del *ground truth* conllevaría la obtención de unos resultados más precisos y la posibilidad de realizar experimentos más completos.
El desarrollo de una aplicación sobre el sistema debe ser meditado con anterioridad, ya que dependiendo de la aplicación elegida habrá que cambiar partes del sistema en consecuencia (reajustar parámetros, implementar nuevas características). Algunos ejemplos de aplicaciones que podrían implementarse sobre el sistema serían detección de eventos inusuales o análisis del comportamiento de los objetos detectados.