UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Departamento de Automática

Vision-based Traffic Monitoring System with Hierarchical Camera Auto-Calibration

PhD Thesis

Author

Sergio Álvarez Pardo

Supervisors

Dr. D. Miguel Ángel Sotelo Vázquez
Dr. D. David Fernández Llorca

2013

Resumen

En las últimas décadas, el tráfico, debido al aumento de su volumen y al consiguiente incremento en la demanda de infraestructuras de transporte, se ha convertido en un gran problema en ciudades de casi todo el mundo. Constituye un fenómeno social, económico y medioambiental en el que se encuentra inmersa toda la sociedad, por lo que resulta importante tomarlo como un aspecto clave a mejorar. En esta línea, y para garantizar una movilidad segura, fluida y sostenible, es importante analizar el comportamiento e interacción de los vehículos y peatones en diferentes escenarios. Hasta el momento, esta tarea se ha llevado a cabo de forma limitada por operarios en los centros de control de tráfico. Sin embargo, el avance de la tecnología, sugiere una evolución en la metodología hacia sistemas automáticos de monitorización y control.

Este trabajo se inscribe en el marco de los Sistemas Inteligentes de Transporte (ITS), concretamente en el ámbito de la monitorización para la detección y predicción de incidencias (accidentes, maniobras peligrosas, colapsos, etc.) en zonas críticas de infraestructuras de tráfico, como rotondas o intersecciones. Para ello se propone el enfoque de la visión artificial, con el objetivo de diseñar un sistema sensor compuesto de una cámara, capaz de medir de forma robusta parámetros correspondientes a peatones y vehículos que proporcionen información a un futuro sistema de detección de incidencias, control de tráfico, etc.

El problema general de la visión artificial en este tipo de aplicaciones, y que es donde se hace hincapié en la solución propuesta, es la adaptabilidad del algoritmo a cualquier condición externa. De esta forma, cambios en la iluminación o en la meteorología, inestabilidades debido a viento o vibraciones, oclusiones, etc. son compensadas. Además el funcionamiento es independiente de la posición de la cámara, con la posibilidad de utilizar modelos con *pan-tilt-zoom* variable para aumentar la versatilidad del sistema.

Una de las aportaciones de esta tesis es la extracción y uso de puntos de fuga (a partir de elementos estructurados de la escena), para obtener una calibración de la cámara sin conocimiento previo. Esta calibración proporciona un tamaño aproximado de los objetos buscados, mejorando así el rendimiento de las siguientes etapas del algoritmo. Para segmentar la imagen se realiza una extracción de los objetos móviles a partir del modelado del fondo, basándose en mezcla de Gaussianas (GMM) y métodos de detección de sombras. En cuanto al seguimiento de los objetos segmentados, se desecha la idea tradicional de considerarlos un conjunto. Para ello se extraen características cuya evolución es analizada para conseguir finalmente una agrupación óptima que sea capaz de solventar oclusiones.

El sistema ha sido probado en condiciones de tráfico real sin ningún conocimiento previo de la escena, con resultados bastante satisfactorios que muestran la viabilidad del método.

Abstract

In recent decades, traffic has become a great problem in most of the cities around the world, due to the increment of the number of vehicles and the transport infrastructures demand. It represents a social, economical and environmental phenomenon which involves all the society. Therefore it is crucial to consider it as a key area to improve. Along these lines, and to guarantee a safe, fluid and sustainable mobility, it is important to analyse the behaviour and interaction of vehicles and pedestrians in different scenarios. Not long ago this task was performed only by human operators at traffic control centres. However, the advances in technology, suggest an evolution in the methodology towards the automation of the surveillance and control.

The presented work describes a target detection system on transport infrastructures, for applications in the framework of Intelligent Transportation Systems (ITS). Particularly as a monitoring system to detect and predict incidents (traffic accidents, dangerous manoeuvres, traffic jams, etc.) on critical areas of transportation infrastructures, like intersections or roundabouts. To achieve this objective, a monocular vision-based approach with hierarchical camera auto-calibration is proposed. It is able to measure parameters of vehicles and pedestrians, as an input of a future incident detection system, traffic control system, etc.

The common problem of computer vision in this kind of applications, and where the proposed solution puts special emphasis, is the adaptability of the algorithm to external conditions. Accordingly, illumination or weather changes, occlusions, instabilities due to wind or vibrations, etc. are compensated. Furthermore the algorithm is independent of the position of the camera, and it is able to work with variable *pan-tilt-zoom* cameras in fully self-adaptive mode.

One of the contributions of this thesis is the extraction and use of vanishing points, through structured elements of the image, to obtain an automatic calibration of the camera without any prior knowledge. This calibration provides an approximate size of the searched targets, improving the performance of the detection steps. To segment the image, a background subtraction method, based on Gaussian Mixture Models (GMM), image stabilization and shadow detection algorithms, is used. Finally about tracking, the traditional idea of considering objects as a whole is rejected. Instead, characteristic target features are extracted and analysed to achieve an optimal clustering which deals with occlusions.

In the document, the results obtained in real traffic conditions are presented and discussed, without any prior knowledge of the scene or the camera.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent decades, traffic has become a great problem around the world, due to the increment of the number of vehicles and the transport infrastructures demand. It represents a social, economical and environmental phenomenon which involves all the society. To make the governments aware of its seriousness, the European Commission devoted the year 2011 in their annual *White Paper* to report about the importance to improve and build a competitive transport system[1]. They assume that new technologies for vehicles and traffic management will be key to achieve the goal: "*The race for sustainable mobility is global. Delayed action and timid introduction of new technologies could condemn the EU transport industry to irreversible decline. Overall, transport infrastructure investments have a positive impact on economic growth, create wealth and jobs, and enhance trade, geographical accessibility and the mobility of people*".

Along these lines, and to guarantee a safe, fluid and sustainable mobility, it is important to analyse the behaviour and interaction of vehicles and pedestrians in different scenarios. Not long ago this task was performed only by human operators at traffic control centers, like the center shown in Figure 1.1. However, it is demonstrated that the level of attention and the human accuracy of incident detection decreases over time, and it is becoming a harder task because of the increment in the number of surveillance cameras, transport infrastructures and vehicles. The advances in technology, suggest an evolution in the methodology towards the automation of the surveillance and control.

The associated technologies which could be automated can vary from basic management systems, such as traffic signal control, variable message signs or infrastructure-to-vehicle communication; to monitoring applications, such as automatic incident detection (accidents, wrong way vehicles, stopped vehicles on the road, etc.), vehicle counting, congestion detection, plate number recognition, speed cameras or even tunnel surveillance or security CCTV systems.

In the framework of Intelligent Transportation Systems (ITS), there are several existing technologies to address some of these issues. On the one hand, intrusive systems like inductive loop detectors (ILDs), radar and laser. ILDs have been used

[1] European Commission White paper 2011. Roadmap to a Single European Transport Area - Towards a competitive and resource efficient transport system.

Figure 1.1: Example of traffic control center.

extensively, providing good detection and classification results. However they have significant drawbacks: (i) their use involves the excavation of the road to place the sensing devices, (ii) they are installed per lane so vehicles travelling between lanes are miss-detected, and (iii) they cannot manage well traffic congestions. Technologies based on time-of-flight sensors can deliver similar counting and classifying results, but as well they have important drawbacks: Radar has high error on horizontal resolution and it is extremely complex to interpret, and Laser is more expensive and does not work under certain weather conditions. Moreover the emission of radiation of both systems must not be forgotten.

On the other hand, computer vision analysis systems have become popular in transport management due to their capability to extract very rich information on road traffic (shape, texture, color), track a variable number of targets and classify them. Considering that and adding a lower price, no intrusion and easier installation and maintenance, video processing seems to be the best alternative to the technologies cited before; to detect and track vehicles and pedestrians for traffic flow estimation, signal timing, safety applications or video surveillance. Furthermore, there are many cameras already installed on the roadside, particularly at intersections and roundabouts, so part of the installation work is already done.

In order to extend the surveillance area and overcome occlusions, there is a possibility of using multiple cameras in the same infrastructure. However, the higher price and complexity of the installation, the computational cost, etc. make the monocular system the best solution; cheap, versatile and flexible.

1.2 Visual surveillance of transportation infrastructures

Recently, a lot of research has been carried out on systems to detect and track vehicle and pedestrians using vision from traffic infrastructures. Nevertheless very few address the problems of complex urban environments, the adaptability to every external condition or the chance to vary the position, angle or zoom of the camera in order to make the system as versatile as possible ("plug&play").

As the traffic applications often use fixed cameras, most of the related work is based on the *background subtraction* algorithm. The idea is to subtract the current image from a reference image, which is a representation of the scene background, to find the foreground objects. The technique has been used for years in many vision systems as a preprocessing step, and the results obtained are fairly good. However the algorithm is susceptible to several problems such as sudden illumination changes, cast shadows, camera shake or image noise; which often cause serious errors due to misclassification of moving objects. Moreover, the size of the foreground targets is very dependent of the position of the camera. In the next subsections, the specific challenges related that have to be solved to get a complete application are summarized.

1.2.1 Position of the camera

Before starting to program a computer vision algorithm, one of the first questions to make is related to the size of the searched targets. The fact is how far the camera from the objects is, because the size depends on the distance. In case of surveillance applications, the position of the camera is totally random, and it is different from one infrastructure to another. Therefore, as the goal is to develop a “plug&play” system, the approximate dimensions of the objects are needed.

Camera calibration reveals fundamental to estimate 3D sizes of the targets and makes object detection and tracking more robust to noise and occlusions, and adaptable to every camera location possible. In particular auto-calibration methods seem to be the most suitable way to recover camera parameters for these types of autonomous applications. Since most surveillance systems make use of one single camera, auto-calibration can be only achieved from inherent structures of the scene or structured object motion.

Figure 1.2 shows two traffic surveillance images. The sizes of the searched objects in each system are completely different. Furthermore, due to camera perspective, objects have different sizes depending on its position in the image, and occlusions affect in a different way.

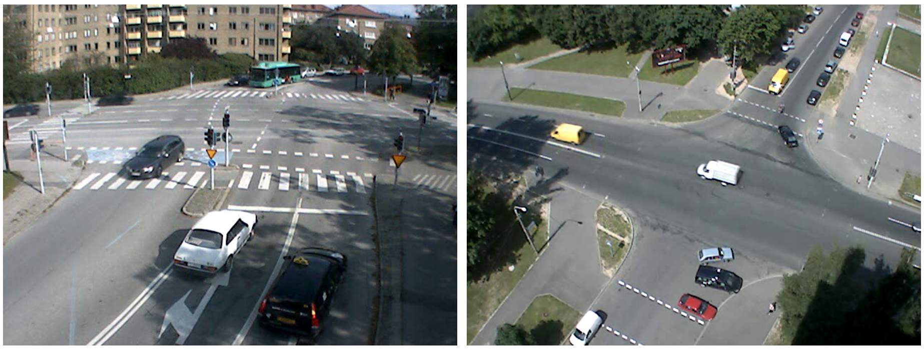


Figure 1.2: Different traffic scenes and searched objects sizes.

1.2.2 Pan-tilt-zoom specifications

In recent years, Pan-tilt-zoom (PTZ) cameras have been widely used for monitoring and surveillance applications. These cameras provide a full coverage for a given area due to the chance to zoom and modify the viewing angles, making the systems robust and flexible. The problem of using PTZ cameras for image processing applications, and in particular in the one presented in this thesis, is the continuous change of scene. The challenge is to detect camera motion and zoom, to adapt the parameters computed for background subtraction and calibration.

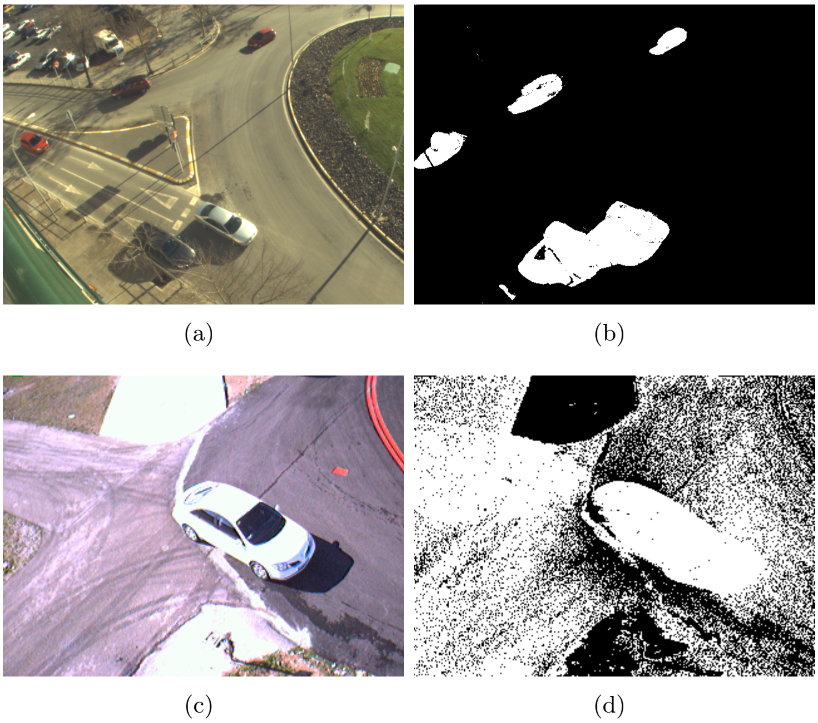
One of the main advantages of the auto-calibration method developed lies in the possibility to change the position and parameters of the camera without manual supervision of the system process.

1.2.3 Object (cast) shadows and illumination changes

Generally, most foreground segmentation methods are sensitive to illumination conditions. Accordingly, cast shadows (occlusion of light sources by foreground objects) and illumination changes due to meteorology, are detectable as foreground since they typically differ from the background. Moreover, shadows are connected to and have the same motion as the objects casting them.

There are several problems associated to these effects: the segmented area can be bigger than expected and merge different objects into one, the shape of the targets can be distorted, new erroneous objects can appear, etc. giving rise to an inaccurate detection and a low tracking performance. For all these reasons, a reliable and accurate method to identify illumination changes is required.

Figure 1.3 depicts examples of shadows and illumination changes which make foreground segmentation more difficult to analyse or invalid.

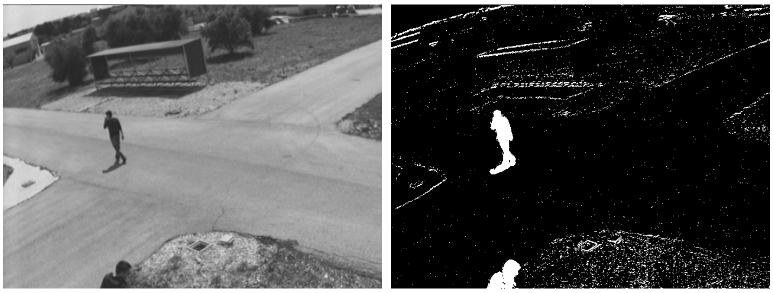

(a) (b)

(c) (d)

Figure 1.3: Background subtraction problems due to cast shadows (a)-(b), and a sudden illumination change (c)-(d).

1.2.4 Instabilities of the camera

Vibrations of the camera structure, small movements due to wind or camera jitter make the captured image unstable. Because of that, the image does not fit to the background model and the result is inaccurate as can be seen in Figure 1.4.



(a) (b)

Figure 1.4: Background subtraction problems due to instabilities of the camera.

1.3 Document structure

After the present introduction, the remainder of the document is organized as follows. Chapter 2 contains a brief review of the most significant published research on traffic infrastructures monitoring. In particular about calibration methods, object segmentation approaches, shadow detection algorithms and tracking works. A discussion and the main objectives of the thesis are then introduced.

Chapter 3 describes the developed camera auto-calibration method, based on vanishing points, and the hierarchical system proposed, with results that prove the viability of the system.

In Chapter 4 the global monitoring approach is presented, whit the segmentation and tracking algorithms used. Results for experiments on real traffic conditions are presented and discussed.

Finally Chapter 5 contains the conclusions and main contributions of the thesis, and future research lines that may spring from it.

Chapter 2

State of the art

Detecting and tracking pedestrians and vehicles in traffic applications has been one of the most active fields of ITS research for the last years. Many approaches try to solve this problem, however, due to the numerous advantages against other methods, monocular vision stands out as the best solution (see Section 1.1). This chapter presents a brief survey of the state of the art in monocular target monitoring on transport infrastructures. For the sake of clarity, the related work will be divided into three sections according to the steps of the algorithm: camera calibration, object segmentation and tracking.

2.1 Camera calibration. Auto-calibration

Camera calibration, is a fundamental stage in computer vision, essential for many applications. The process is the determination of the relationship between a reference plane and the camera coordinate system (extrinsic parameters), and between the camera and the image coordinate system (intrinsic parameters). These parameters are very useful to recover metrics from images or apply prior information of 3D models to estimate 2D pose of targets, making object detection and tracking more robust to noise and occlusions.

The standard method to calibrate a camera is based on a set of correspondences between 3D points and their projections on image plane [1], [2]. However, this method requires either prior information of the scene or calibrated templates, limiting the feasibility of surveillance algorithms in most possible scenarios. In addition, calibrated templates are not always available, they are not applicable for already-recorded videos and if the camera is placed very high their small projection can derive in poor accurate results. Finally in case of having PTZ cameras, using a template each time the camera angles or zoom change is not available. One novel method which solves the problem of the template is the orthogonal calibration proposed by Kim [3]. The system extracts the world coordinates from aerial pictures (on-line satellite images) or GPS devices to make the correspondences with the image captured. However this system is dependent on prior information from an external source and it does not work indoor. Figure 2.1 shows an example of the point extraction and the results of the calibration.

Therefore auto-calibration seems to be the more suitable way to recover camera parameters for surveillance applications. Since most of these applications make use of

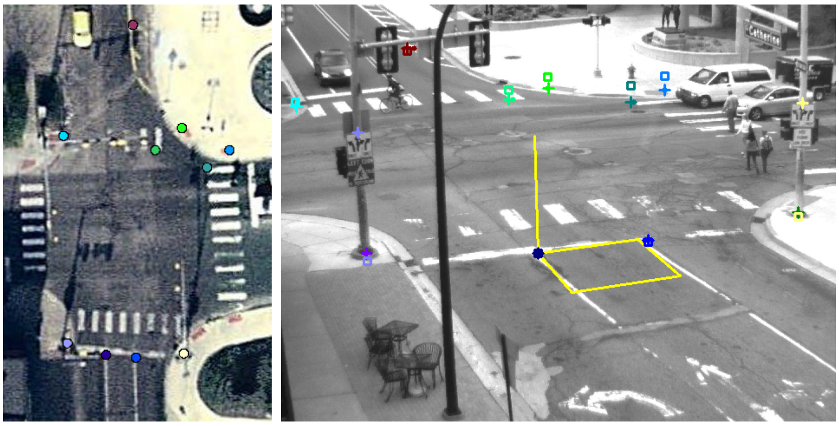


Figure 2.1: Orthogonal calibration by [3].

only one static camera, auto-calibration cannot be achieved from camera motion, but from inherent structures or flow patterns of the scene.

One of the distinguished features of perspective projection is that the image of an object that stretches off to infinity can have finite extent. For example, parallel world lines are imaged as converging lines, which image intersection point is called *vanishing point*. In 1990 Caprile and Torre [4] developed a new method for camera calibration using simple properties of vanishing points. In their work the intrinsic parameters of the camera were recovered from a single image of a cube. In a second step, the extrinsic parameters of a pair of cameras were estimated from an image stereo pair of a suitable planar pattern. The technique was improved by Cipolla et al. [5], who computed both intrinsic and extrinsic parameters from three vanishing points and two reference points from two views of an architectural scene. However these assumptions were incomplete, because as demonstrated by Hartley, Zisserman and Liebowitz in different publications, and summarized in [1], it is possible to obtain all the parameters needed to calibrate a camera from three *orthogonal* vanishing points.

From the works mentioned before, a lot of research has been done to calibrate cameras in architectural environments (Rother [6], Tardif [7], etc...). All these methods are based on scenarios where the large number of orthogonal lines provide an easy way to obtain the three orthogonal vanishing points, just taking the three main directions of parallel lines. Examples of architectural scenarios and the main orthogonal lines extracted are depicted on Figure 2.2.

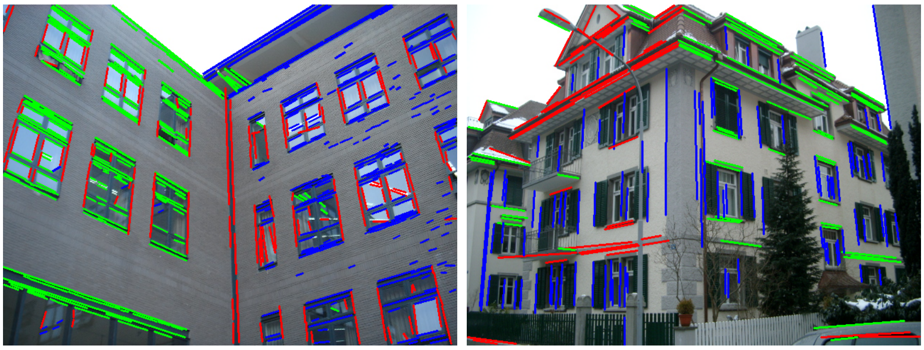
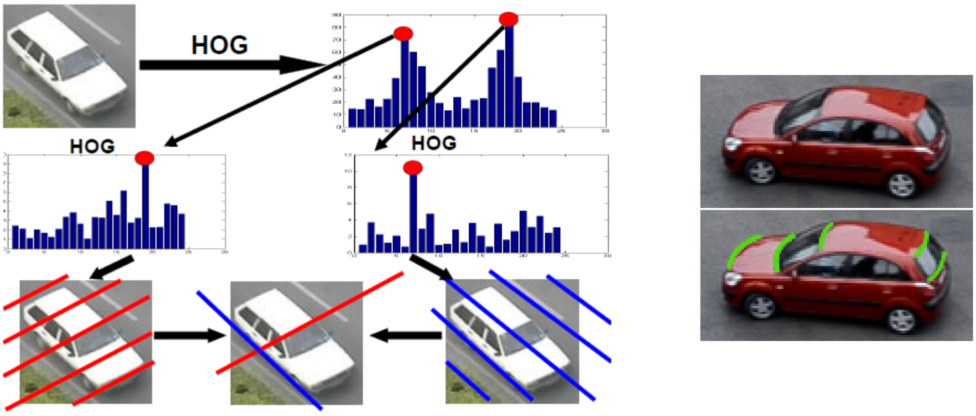


Figure 2.2: Architectural scenarios and main orthogonal lines by [6].

Nevertheless, in absence of so strong structures, as usual in the case of traffic scenes, the vanishing point-based calibration is not applicable. In this context, a different possibility is to make use of object motion. The complete camera calibration work using this idea was introduced in 2006 by Lv et al [8]. The method uses a tracking algorithm to obtain multiple observations of a person moving around the scene; computing the three orthogonal vanishing points by extracting head and feet positions in their leg-crossing phases. The approach requires accurate localization of these positions, which is a challenge in traffic surveillance videos. Furthermore, the localization step uses FFT based synchronization of a person's walk cycle, which requires constant velocity motion along a straight line. Finally it does not handle noise models in the data and assumes constant human height and planar human motion, so the approach is really limited. Based on this knowledge, Junejo proposed a quite similar calibration approach for pedestrians walking on uneven terrains in [9]. There are no restrictions as with Lv's work, but the intrinsic parameters are estimated by obtaining the infinite homography from all the extracted points in multiple cameras.

To manage the inconveniences shown in the previous paragraph, the solution lies in computing the three vanishing points by studying three orthogonal components with parallel lines in the moving objects or their motion patterns. Zhang et al. [10] presented a self-calibration method using the orientation of pedestrians and vehicles. The method seems to extract a vertical vanishing point from the main axis direction of the pedestrian trunk, perpendicular to the ground plane. Additionally, two horizontal vanishing points are extracted by analysing the histogram of oriented gradients of moving cars, as shown on Figure 2.3(a). The idea is interesting and it was initially implemented for this thesis. However, the straight vehicles used by Zhang differ from the modern ones, usually with more irregular and rounded shapes (Figure 2.3(b)). Finally, the pedestrian detection step is not described and results are not depicted in the paper.



(a)

(b)

Figure 2.3: Differences between modern and old vehicles in terms of HOG. (a) Zhang flowchart for estimation of line equations for vehicles [10]. (b) Example of modern vehicle an its "perpendicular" lines

Hodlmoser et al. present a different approach [11]. They use zebra-crossings with known measures to obtain the ground plane, and pedestrians to obtain the vertical lines. The problem is the maximum distance the camera can be from the ground and the necessity of knowing real distances from the scene.

The mentioned works are summarized and grouped in Table 2.1, where a taxonomy of approaches is shown.

A. **Number of cameras**

   a. 1 camera (*Hartley, 2000 [1]; Tsai, 1986 [2]; Kim, 2009 [3]; Rother, 2002 [6]; Tardif, 2009 [7]; Lv, 2006 [8]; Zhang, 2011 [10]; Hodlmoser, 2010 [11]*).

   b. Multiple cameras (*Caprile, 1990 [4]; Cipolla, 1999 [5]; Junejo, 2009 [9]*).

B. **Technique**

   a. 3D points correspondence (*Hartley, 2000 [1]; Tsai, 1986 [2]; Kim, 2009 [3]*).

   b. Vanishing points (*Caprile, 1990 [4]; Cipolla, 1999 [5]; Rother, 2002 [6]; Tardif, 2009 [7]; Lv, 2006 [8]; Junejo, 2009 [9]; Zhang, 2011 [10]; Hodlmoser, 2010 [11]*).

C. **Needed scenario elements**

   a. Calibration pattern (*Hartley, 2000 [1]; Tsai, 1986 [2]*).

   b. Cube (*Caprile, 1990 [4]*).

   c. Architectural (*Cipolla, 1999 [5]; Rother, 2002 [6]; Tardif, 2009 [7]*).

   d. Urban (*Kim, 2009 [3]; Lv, 2006 [8]; Junejo, 2009 [9]; Zhang, 2011 [10]; Hodlmoser, 2010 [11]*).

D. **Prior knowledge**

   a. Pattern measurements (*Hartley, 2000 [1]; Tsai, 1986 [2]*).

   b. GPS coordinates (*Kim, 2009 [3]*).

   c. Camra height (*Kim, 2009 [3]; Caprile, 1990 [4]; Cipolla, 1999 [5]; Rother, 2002 [6]; Tardif, 2009 [7]; Zhang, 2011 [10]*).

   d. Pedestrians height (*Lv, 2006 [8]; Junejo, 2009 [9]*).

   e. Metrics from the scene (*Hodlmoser, 2010 [11]*).

E. **Restrictions**

   a. Orthogonal structures (*Cipolla, 1999 [5]; Rother, 2002 [6]; Tardif, 2009 [7]*).

   b. Constant and straight motion (*Lv, 2006 [8]*).

   c. Straight vehicles (*Zhang, 2011 [10]*).

   d. Camera height (*Hodlmoser, 2010 [11]*).

   e. None (*Hartley, 2000 [1]; Tsai, 1986 [2]; Kim, 2009 [3]; Caprile, 1990 [4]*).

Table 2.1: Taxonomy of camera calibration approaches.

In this thesis, the author proposes a self-calibration procedure based on vanishing points through an hierarchical process, which covers most of traffic infrastructure scenarios with all possible configuration. The objective is to obtain both intrinsic and extrinsic camera parameters without any restriction in terms of constraints (e.g. restrictions mentioned in previous paragraphs, vehicles driven in only one road direction [12], deprecated camera roll [13], etc.) or the need of prior information.

2.2 Object segmentation

Traffic surveillance systems consist on detecting and tracking moving objects by a static camera. In this context, several segmentation approaches have been proposed, although *background subtraction* based ones are the most discussed pixel-wise techniques. This method requires a relatively small computation time and shows robust detection in good illumination conditions. However, it suffers under the presence of shadows and sudden illumination changes, distorting the estimation of the targets. The object segmentation methodologies have then two main discussion points, in which the section is divided: *background subtraction* and *cast shadows and illumination changes detection.*

2.2.1 Background subtraction

The basic idea of the method is to subtract the current image from a reference image that models the background scene, to obtain the moving objects. In spite of the lot of research done in the field, many difficulties have still to be considered, especially under dynamic environments, changing situations, different weather conditions, etc.

Most researches have abandoned non-adaptive methods because of the need of manual initialization. Without an update of the background model, errors accumulate over time and do not allow changes in the scene. Therefore they are only useful in highly supervised and short-term applications. A background subtraction method should continuously estimate a statistical model of the variation for each pixel. A common used method is averaging the images over time, creating a background approximation which is similar to the current static scene except where motion occurs. While this is effective in situations where objects move continuously and the background is visible a significant portion of the time, it is not robust to scenes with many moving objects, particularly if they move slowly. It also recovers slowly when the background is uncovered, and has a single and predetermined threshold for the entire scene.

The existing adaptive background modelling methods can be classified as either single-layer or multi-layer. Single layer methods obtain a model for the color distribution of each pixel, based on past observations. Usually, a single Gaussian is used to model the statistical distribution of a background pixel, being updated through a blending approach. These models are fast and simple, but in practice, multiple surfaces often appear in the view of a particular pixel and they are not able to adapt to multiple backgrounds. Figure 2.4 shows the importance of using adaptive methods, because of the movement of the sun, and multiple layers because of the movement of the trees.

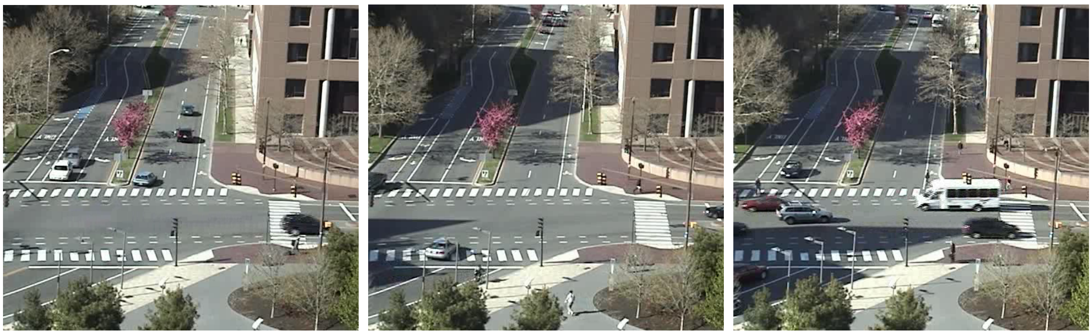


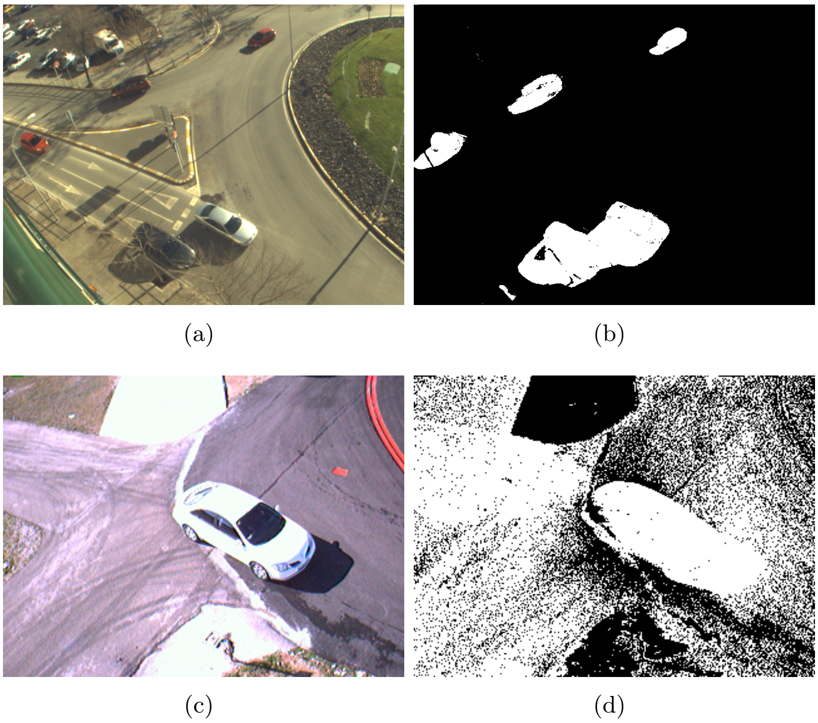
Figure 2.4: Shadows moving in the same scene over time.

The use of *Gaussian Mixture Models* (GMM) has enjoyed tremendous popularity since it was first proposed for background modelling by Friedman and Rusell [14] in 1997. After them, in 1999 Stauffer and Grimson [15] developed more efficient update equations and wrote a very important publication base of many works. In 2005, Lee extended the standard equations to increase the adaptation speed of the model [16]; and finally in 2006 Zivkovic [17] improved the method incorporating a model selection criterion to choose the proper number of components for each pixel on-line. This segmentation is very robust to variations in the scene due to progressive changes of the illumination, moving tree leaves, slow moving objects, etc. Nevertheless, it is vulnerable to sudden illumination changes with just a few Gaussians (usually 3 to 5), so a non-parametric technique was developed for estimating background probabilities at each pixel from many recent samples over time using Kernel density estimation [18]. The problem is the computational cost and the difficulty of choosing a correct kernel and kernel size.

In conclusion, due to a good compromise between robustness and performance, GMM is the most used technique for background subtraction. There are drawbacks with illumination changes, cast shadows and camera shake; but it gives a compact model useful for further postprocessing. In this thesis a robust adaptive background segmentation method, similar to the Zivkovic's one, is used.

2.2.2 Cast shadows and illumination changes

Despite its success, the mixture of gaussians method fails, as said before, classifying pixels as foreground or background in case of sudden illumination changes or cast shadows (Figure 2.5).



(a) (b)

(c) (d)

Figure 2.5: GMM Background subtraction problems due to cast shadows (a)-(b), and sudden illumination changes (c)-(d).

Earlier systems set manual thresholds or used supervised algorithms to avoid these problems. But recently many works have been published looking for a solution. According to the taxonomy proposed by Prati in [19] (see Figure 2.6), the algorithms can be classified as deterministic and statistical; and the first one subdivided into model-based and property-based. On the one hand, model-based methods use prior knowledge of scene geometry, target objects or light sources to predict and remove shadows. Joshi et al. [20] propose an algorithm which detects shadows by using Support Vector Machine (SVM) and a shadow model, learned and trained from a database. Reilly et al. [21] propose a method based on a number of geometric constraints obtained from meta-data (latitude, longitude, altitude, as well as pitch, yaw and roll). Specifically, they obtain the orientation of ground plane normal, the orientation of cast shadows in the scene, and the relationship between human heights and the size of their corresponding shadows. The problem of these methods is that they need prior information and offline processing.

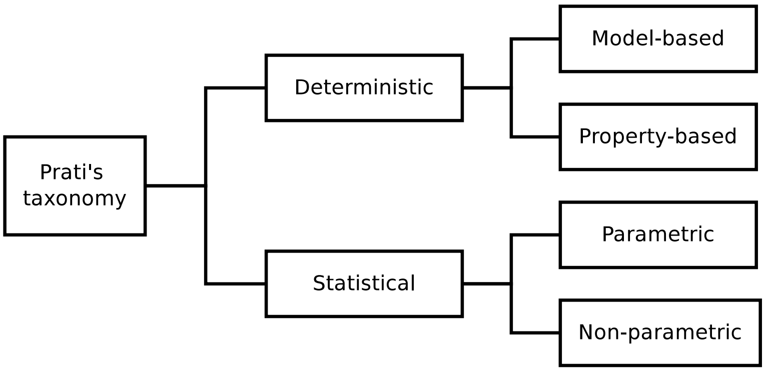


Figure 2.6: Classification of shadow detection methods according to Prati [19].

On the other hand, property-based approaches use features like geometry, brightness or color to detect illumination changes. Some authors detect shadows in grayscale video sequences. Jacques et al. [22] use the normalized cross-correlation as an initial step for shadow detection, and refine the process using local statistics of pixel ratios. However, these algorithms sometimes become invalid since pixels of different colors may have a similar gray level. Therefore, most authors detect illumination changes using color information. Atev et al. [23] integrate an illumination filter before processing the image. The filter scales and adds an offset to all pixels in the image to prevent sudden brightness or contrast changes while preserving the color information. The method seems to be effective to compensate sharp changes in the overall scene, but it does not work well with local changes. Horprasert et al. [24] propose a color model to classify each pixel as foreground, background, shadowed background, or highlighted background. The algorithm performs well in indoor environments or under certain illumination conditions, but not for the variability of traffic scenes. Salvador et al. [25] use the idea that a shadow darkens the surfaces on which it is cast, to identify an initial set of shadowed pixels, that is then pruned by using color invariance and geometric properties of shadows. In [26], Cucchiara et al. use the hypothesis that shadows reduce surface brightness and saturation while maintaining hue properties in the HSV color space. Schreer et al. [27] adopt the YUV color space to avoid using the time consuming HSV color transformation and segment shadows from foreground objects based on the observation that shadows reduce the YUV pixel value linearly. These methods are deterministic approaches which can deal with illumination noises and soft shadows but they fail representing heavily shadows where color and chromaticity information are totally lost.

Statistical models can be subdivided into parametric and non-parametric. Parametric approaches use a series of parameters which determine the characteristics of the statistical functions of the model, while non-parametric automate the selection of the model parameters as a function of the observed data during training. There are also statistical approaches such as [28] that uses Gaussian Mixture Model to describe moving cast shadows, or [29] which models shadows using multivariate Gaussians. These methods can adapt to changing shadow conditions and provide a low number of false detections. However, the hypothesis is not effective with soft shadows and if the shadowed pixels are seldom or they have never been taken up by the algorithm.

In 2012, Sanin et al. published a survey [30] to evaluate newer shadow detection algorithms. They classify the reviewed works into four categories: chromacity-based methods, geometry-based methods, texture-based methods and physical methods (see Figure 2.7). Chromacity-based techniques do not provide new ideas and have the same advantages and disadvantages mentioned before. About geometry-based approaches, all publications exposed are focused on pedestrians or on vehicles but not on both or any object in general, which is one of the objectives of a surveillance system. Texture-based works are divided into small region (SR) and large region (LR) approaches. On the one hand the most representative method presented as SR is based on Gabor functions [31]. Region-level correlation is more robust than pixel-level correlation and Gabor functions can provide optimal joint localisation in the spatial/frequency domains. The texture analysis is performed by projecting a neighbourhood of pixels onto a set of Gabor functions with several bandwidths, orientations and phases, and the matching between frame and background is found using Euclidean distance. The problem of small regions is that they are not guaranteed to contain significant textures. In a different paper, Sanin et al. [32] proposed using colour features to create large candidate shadow regions (ideally containing whole shadow areas), which are then discriminated from objects using gradient-based texture correlation.

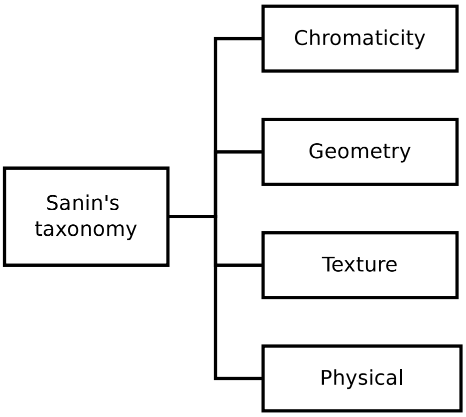


Figure 2.7: Classification of shadow detection methods according to Sanin [30].

Probably the most interesting part of the survey is related to physical methods. The authors chose the work of Huang et al. [33] which does not make prior assumptions about the light sources and ambient illumination. For a pixel, given the vector from shadow to background value, the colour change is modelled using a 3D colour feature, constructed by the illumination attenuation and the direction of the shadow vector. This colour feature describes the appearance variation induced by the blocked light sources on shaded regions.

In spite of the descriptions and results, the survey is not really representative for traffic surveillance applications. The paper provides a graph (Figure 2.8) which exposes a comparison of the analysed methods. For highway tests the performance obtained is too low. Moreover, a source code of the implemented algorithms is available to download and it has been tested with unsatisfactory results. Figure 2.9 shows the foreground mask obtained from different shadow detection algorithms. Shadowed areas are not removed.

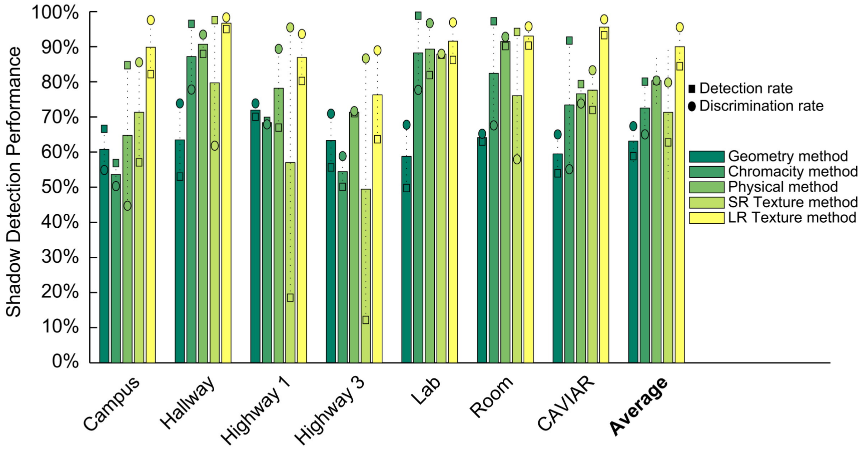


Figure 2.8: Sanin's shadow detection performance graph [30].

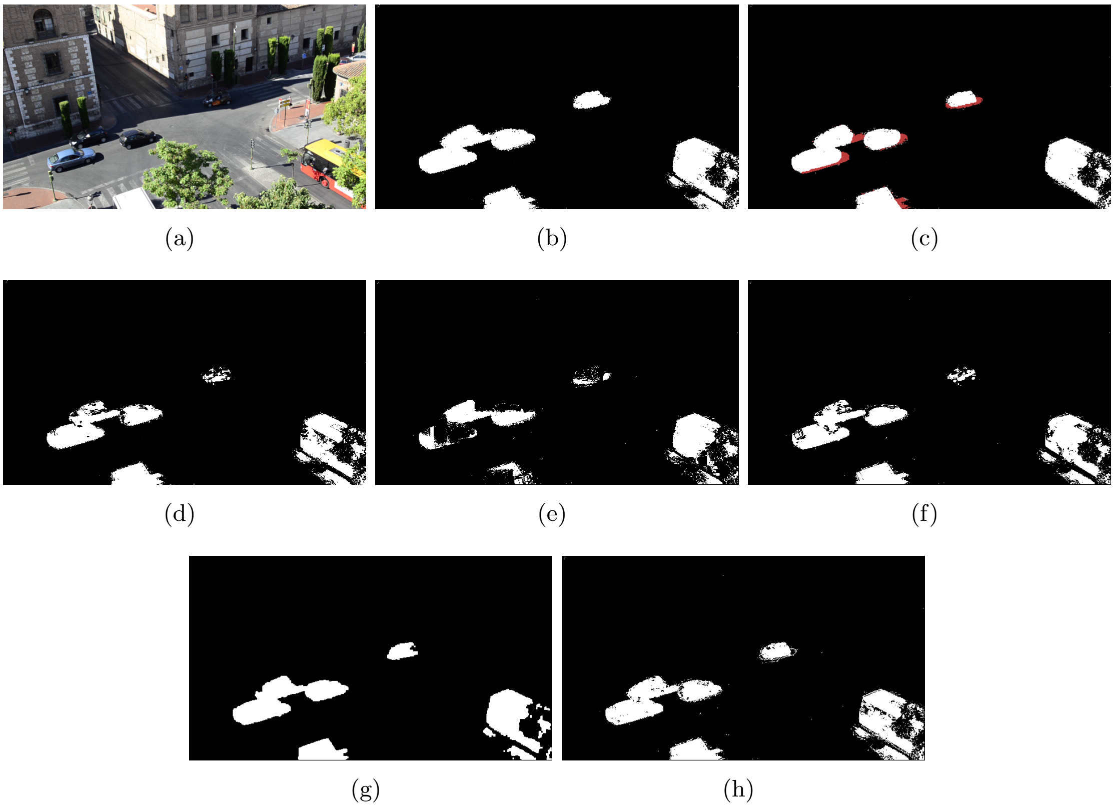

(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)

Figure 2.9: Results of Sanin's survey shadow detection results. (a) Source image. (b) Initial foreground mask. (c) Groundtruth with shadows in red. (d) Foreground after chromacity shadow detection. (e) Foreground after geometric shadow detection. (f) Foreground after SR shadow detection. (g) Foreground after LR shadow detection. (h) Foreground after physical shadow detection.

The main conclusion is that only the simplest methods are suitable for generalisation, but in almost every particular scenario the results could be significantly improved by adding assumptions. As a consequence, there is no single robust shadow detection technique and it seems better for each particular application to develop its own technique according to the nature of the scene.

An overview of the described methods is presented in Table 2.2.

Category	**Method**	**Technique**	**Prior info.**
Model	Joshi [20]	SVM	Offline learned shadows
	Reilly [21]	Orientation of shadows	Information of the Sun and camera
Property	Jacques [22]	Gray NCC	None
	Atev [23]	Image filter	Filter thresholds
	Horprasert [24]	Color model	None
	Salvador [25]	Geometric constrains	Geometric model of targets
	Cucchiara [26]	Color model	None
	Schreer [27]	Color model	None
Parametric	Brissom [28]	GMM	None
	Porikli [29]	GMM	None
Texture	Leone [31]	Gabor functions	None
	Sanin [32]	Gradient-based texture correlation	None
Physical	Huang [33]	Physical illumination properties	None

Table 2.2: Overview of the described shadow detection methods.

2.3 Target extraction and tracking

After detecting moving objects by background subtraction and lighting analysis, there are new challenges for the system. One problem is how to manage occlusions and merged foreground blobs to extract the different targets on the scene. The second one is about the way to track these targets. Four main approaches have been used to solve the problems: 3D model-based, region-based, contour-based and feature-based methods.

There are some methods which address (avoid) the problem of occlusions by locating the camera above the scene. However this condition is not easy to implement, especially in traffic applications, so it will be discarded.

2.3.1 3D model-based methods

The basic principle of these methods is to define a 3-D geometric model as a matching template that describes the shape of the target. They exploit the a priori knowledge of typical objects to localize and recognize vehicles and pedestrians in the scene. This allows the recovering of trajectories with high accuracy for a small number of objects, and even to address the problem of partial occlusions. The weakness is the reliance on detailed geometric object models. It is unrealistic to expect being able to have models for all targets that could be found in a real scenario. An example of a 3D model-based method is the work of Dahlkamp et al. in [34], with a result depicted in Figure 2.10.

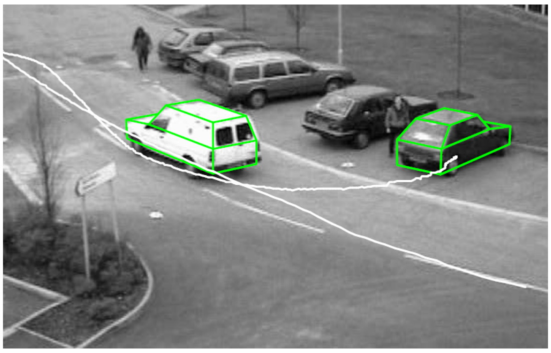


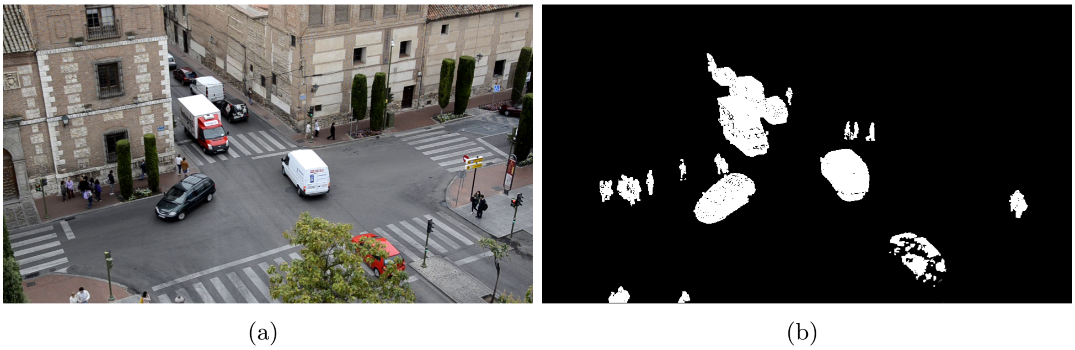
Figure 2.10: Example of 3D model-based method by [34].

2.3.2 Region-based methods

The idea of region or blob-based methods is to identify connected regions of the image (blobs) which represent the targets searched. Regions can be obtained through background subtraction, and then tracked over time using information provided by the entire region (motion, size, color, shape, texture, centroid). Many approaches use Kalman filters for that purpose.

Region-based tracking is computationally efficient and works well in free-flowing traffic. However, under partial occlusions, crowds or congested traffic conditions, blobs can be merged making the task of segmenting individual targets very difficult. These methods cannot usually cope with complex deformation or a cluttered background.

Figure 2.11 shows a simple example where vehicles are partially occluded and appeared merged in a large blob. Moreover, in the case of the red car in the bottom part of the image, the tree splits its mask into several parts. Region-based methods are unable to manage these situations.



(a)

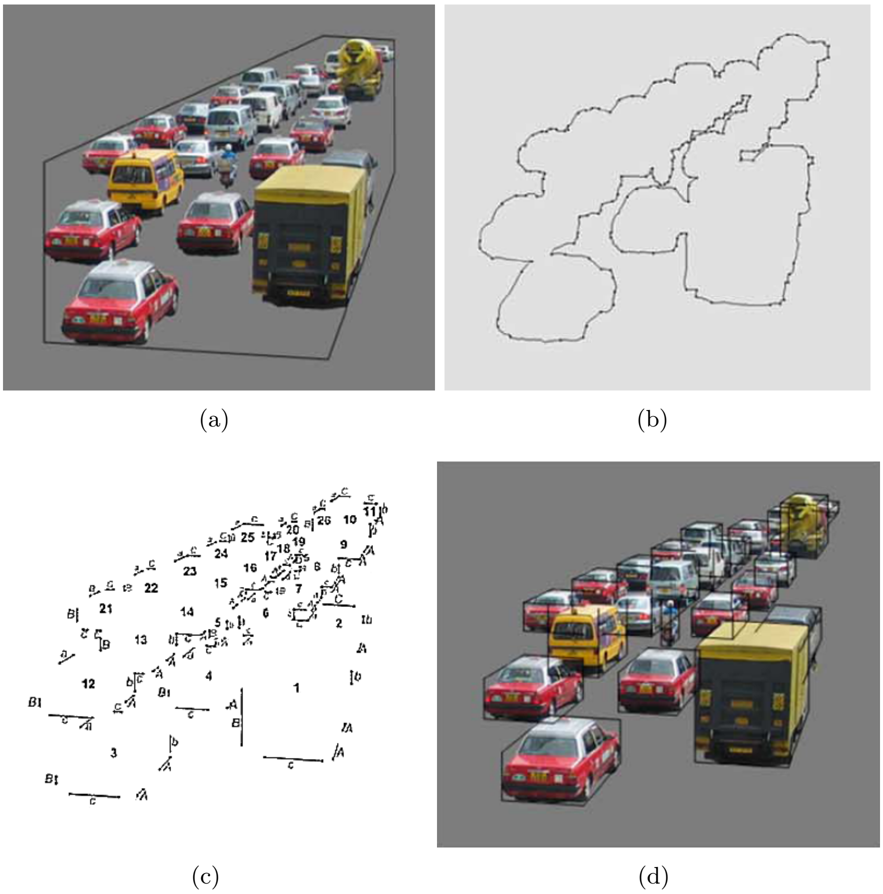
(b)

Figure 2.11: Example of blob merge due to partial occlusions and blob split due to a tree, after applying background subtraction. (a) Original image. (b) Foreground mask.

2.3.3 Contour-based methods

Similarly to the previous subsection, contour-based methods use active contours to model the boundary of vehicles, which can be updated dynamically. They have relatively lower computational complexity, but the initialization for each target is a complex issue, which is difficult to achieve efficiently in practice. In some cases, shape restriction is applied together with a Kalman filter to estimate the spatio-temporal relationships of the contour to improve tracking robustness in the presence of occlusion, as it is done by Fan [35]. Experimental results were encouraging for tracking vehicles that are well-separated but the inability to segment partially occluded objects remains.

The only work found with interesting results, based on contour analysis, is the proposed by Pang et al. in [36]. The method first deduces the number of vertices per individual vehicle from the camera configuration. Next a contour description model is used to describe the contour segments direction regarding its vanishing points. Finally, it assigns a resolvability index to each occluded vehicle based on a resolvability model, from which each occluded vehicle model is resolved and the vehicle dimension is measured. As can be seen on Figure 2.12 results are impressive. However the system only works with a straight and common motion pattern and has not been tested with pedestrians (with a more variable contour model). Therefore the method is not valid for intersections, roundabouts and other scenarios needed by a complete traffic surveillance system.



(a) (b)

(c) (d)

Figure 2.12: Example of occlusion management by Pang et al. method [36]. (a) Original image. (b) External contour extraction. (c) Computed contours. (d) Final result.

2.3.4 Feature-based methods

Feature-based methods give up the idea of tracking objects as a whole. Instead, they track features extracted from the targets. Even in the presence of partial occlusion, some of the features of the moving objects remain visible, so it is a chance to overcome the problem. Furthermore, the same algorithm can be used for tracking in daylight, twilight or night-time conditions, as well as different traffic conditions and camera positions. It is self-regulating because it selects the most salient features under the given conditions (e.g. window corners, bumper edges... during the day and tail lights at night).

In stereo-based methods, two or more images of the same scene are used to track densely populated objects. Otsuka and Mukawa [37] modeled the spatial structure of the occlusion process between human and its uncertainty, and formulated a recursive Bayesian estimation method for human position and posture. Although they have tested their model using six cameras on five people successfully, its reliance on a large number of camera views implies high computational complexity, which is a big drawback for traffic surveillance.

In a monocular case, the algorithms can adapt successfully and rapidly, allowing real-time processing and tracking of multiple objects in dense traffic. Kanhere et al. [38] use the background subtraction result to segment and track vehicles at low camera angles. They estimate the 3D height of vehicle features by assuming that the bottom of the foreground region is the bottom of the object, but this assumption is only valid in case of no occlusions. Moreover, the vehicle is interpreted as a box, which features lie on one of the four surfaces orthogonal to the road plane. Notwithstanding even at low camera angles there are features in the ceiling and hood so the idea is questionable. The main problem of feature-based methods is how to group the multiple features obtained to separate different objects. An example of commonly used cues for grouping are spatial proximity and similar motion.

One of the most representative works in this area is [39], by Kim. He introduced a dynamic multi-level feature grouping algorithm that can handle various sizes of objects, and provides a set of robust trajectories (Figure 2.13). Although promising results are presented in several transportation scenarios, the system is not fully autonomous. The individual probability distributions for the feature membership are estimated by using a semi-supervised learning procedure. First the algorithm is ran with manually-assigned parameters to obtain a reasonable result. In addition, a user interface is developed to correct observably erroneous membership assignments.

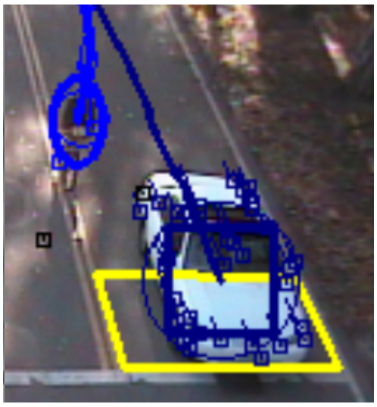


Figure 2.13: Feature tracking by Kim's method [39].

In this thesis a feature-based algorithm is presented. Features are extracted from background subtraction algorithm and tracked by optical flow analysis and an interesting approach based on *flock of features*. After that, a novel 3D clustering method based on occlusion reasoning and MeanShift technique is done in order to separate and track every single object in the image.

2.4 Discussion

Previous sections have introduced a number of used and published methods for transport infrastructures surveillance. It is not possible to mention all the existing approaches, but it has been shown the high diversity of works and the importance of the topic. Some of the presented methods have provided inspiration for the development of the thesis, although their drawbacks have been analysed to contribute with some improvements.

Several conclusions can be extracted from the review of the state of the art:

- Camera calibration is a fundamental stage in this framework, making object detection and tracking more robust to noise and occlusions. The standard methods, based on calibration patterns, limit the feasibility of surveillance algorithms in most possible scenarios, and any non-automatic approach makes the system vulnerable to unexpected variables and dependent on the user interaction. Therefore auto-calibration reveals as the optimum solution. Nevertheless, there are no complete automatic methods published (with single camera and no prior knowledge) which cover many possible situations in a traffic scenario.

- Traffic surveillance systems consist on detecting and tracking moving objects by a static camera. In this context, background subtraction is the most discussed pixel-wise technique because it requires a relatively small computation time and shows robust detection in good illumination conditions. However, it suffers under the presence of shadows and sudden illumination changes, distorting the estimation of the targets. In spite of the lot of research done in the field, many difficulties have still to be considered, especially under dynamic environments, changing situations, different weather conditions, etc.

- The main conclusion about shadow detection algorithms is that only the simplest methods are suitable for generalisation, but in almost every particular scenario the results could be significantly improved by adding assumptions. Consequently, there is no single robust shadow detection technique and it seems better for each particular system to develop its own method according to the nature of the scene.

- One problem associated to traffic surveillance is the high probability of occlusions and the derived difficulties to extract the different targets of the scene. Because of the multiple possible scenarios and the strong variability of objects (vehicle type and model, sizes due to camera position, etc.), feature-based methods reveal as the solution. Even in the presence of partial occlusion, some of the features of the moving objects remain visible, so it is a chance to overcome the problem. Furthermore, the same algorithm can be used for tracking in daylight, twilight or night-time conditions, as well as different traffic conditions and camera positions. Then, the problems come from the way to cluster them.

- From an application perspective, the main technical challenge is the high diversity of camera views, operating conditions and observation objectives in traffic surveillance. As a consequence, there is an important lack of a common framework and most authors use their proprietary sequences. This condition has generated a large diverse body of work, where it is difficult to perform direct comparison between the proposed algorithms.

2.5 Objectives

After the review of the state of the art, and considering the discussion presented before, the aims of this thesis are the following:

1. To develop an automatic hierarchical camera calibration system based on the scene. The aim is to use the most common elements present in a typical traffic scenario to cover as many situations as possible. The system has to be able to detect PTZ displacements in order to recalibrate the camera if necessary.

2. To develop a monocular vehicle and pedestrian detection system. It has to be able to work in a wide range of environments and conditions without any prior knowledge (only an approximate size of the objects searched).

3. To overcome the common traffic surveillance problems like camera vibrations, lighting variations, shadow effects and object occlusions with automatic self-adapting algorithms.

In summary, the final goal and the main contribution of the thesis is the development of a “plug&play and pan-tilt-zoom” traffic monitoring system, based on a low-cost monocular camera. The research objective is to develop an algorithm able to work in a wide range of environments and conditions without any prior knowledge. The challenge is to be robust to illumination changes, adverse weather conditions, target occlusions, small camera movements and complete variations of its position and zoom.

Chapter 3

Camera auto-calibration

3.1 Introduction

Camera calibration is a fundamental stage in computer vision, essential for many applications. As explained in Section 2.1, the implemented methods of the state of the art are unavailable, do not cover all possible scenes, or need prior information to work correctly. In this thesis, a novel self-calibration procedure based on vanishing points is proposed. The objective is to obtain the camera parameters without any restriction in terms of constraints or the need of prior knowledge, to deal with most traffic scenarios and possible configurations. The developed method is explained in the following sections.

3.2 Camera model

The most common geometric model used to represent a camera is the perspective or pin-hole model [1], shown in the Figure 3.1.

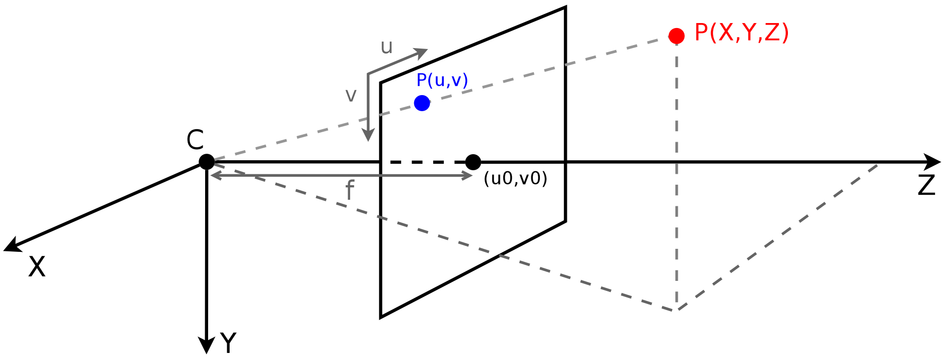


Figure 3.1: Pin-hole camera model.

For a pin-hole camera, perspective projection from the 3D world to an image can be conveniently represented in homogeneous coordinates by the projection matrix P:

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = P \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{3.1}$$

P can be further decomposed by the relative orientation and position of the camera with respect to the world coordinate system, by a 3 x 3 rotation matrix R, a 3 x 1 translation vector T, and the intrinsic parameter matrix K defined by:

$$K = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

where α_u and α_v are scale factors, s is a skew parameter and u_0 and v_0 are the pixel coordinates of the principal point. With the common assumption of zero skew ($s = 0$) and unit aspect ratio ($\alpha_u = \alpha_v = f$ (focal length in pixels)), the global equation has the following form:

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} fR_{11} + u_0R_{31} & fR_{12} + u_0R_{32} & fR_{13} + u_0R_{33} & fT_x + u_0T_z \\ fR_{21} + v_0R_{31} & fR_{22} + v_0R_{32} & fR_{23} + v_0R_{33} & fT_y + v_0T_z \\ R_{31} & R_{32} & R_{33} & T_z \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{3.3}$$

To compute the intrinsic camera parameters and the rotation angles for the camera calibration, the origin of the world coordinate system (WCS) is placed on the ground plane, and it is initially aligned with the camera coordinate system (CCS). Then, it is translated to T, followed by a rotation around the Y-axis by angle $yaw(\alpha)$, a rotation around the X-axis by angle $pitch(\beta)$, and finally, a rotation around the Z-axis by angle $roll(\gamma)$. The corresponding rotation matrix $R = R_z \cdot R_x \cdot R_y$ is formed by:

$$R_z = \begin{bmatrix} cos\gamma & sin\gamma & 0 \\ -sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\beta & sin\beta \\ 0 & -sin\beta & cos\beta \end{bmatrix} \quad R_y = \begin{bmatrix} cos\alpha & 0 & -sin\alpha \\ 0 & 1 & 0 \\ sin\alpha & 0 & cos\alpha \end{bmatrix}$$

Therefore the rotation matrix R is:

$$R = \begin{bmatrix} cos\gamma cos\alpha + sin\gamma sin\beta sin\alpha & sin\gamma cos\beta & -cos\gamma sin\alpha + sin\gamma sin\beta cos\alpha \\ -sin\gamma cos\alpha + cos\gamma sin\beta sin\alpha & cos\gamma cos\beta & sin\gamma sin\alpha + cos\gamma sin\beta cos\alpha \\ cos\beta sin\alpha & -sin\beta & cos\beta cos\alpha \end{bmatrix} \tag{3.4}$$

Real camera lenses typically suffer from non-linear lens distortion which maps straight lines in the image as curved. Pin-hole model is linear and does not manage distortions. However in the case of traffic sequences, the focal distances used are usually long and the analysed objects are far enough from the camera to consider the distortion effect negligible.

The objective of the algorithm is to compute the unknown variables: the focal length f and the rotation angles α, β and γ. It will be achieved extracting three orthogonal vanishing points from the image and using the principal point (u_0, v_0) as the orthocenter of the triangle formed by the three of them (Figure 3.2). This last assumption is widely explained in several references like Zisserman's book [1].

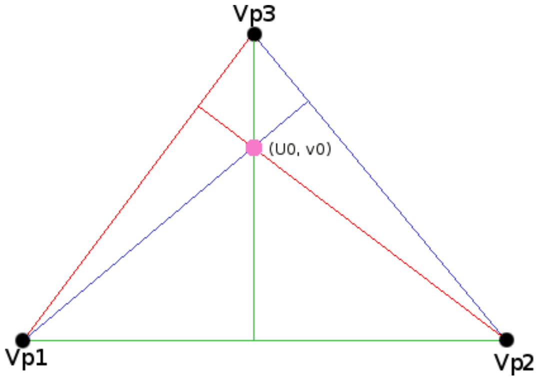


Figure 3.2: For the case of zero skew and unit aspect ratio, the principal point is the orthocenter of an orthogonal triad of vanishing points.

3.3 Camera calibration from vanishing points

One of the distinguishing features of perspective projection is that the image of an object that stretches off to infinity can have finite extent. Particularly, parallel world lines are projected on the image as converging lines, and their image intersection is the vanishing point. A representative example is the effect of the parallel railway tracks shown in Figure 3.3. They never intersect in world coordinates, but have a common point in the image. In the following paragraphs the camera calibration process through three orthogonal vanishing points is analysed. The proposed extraction of these points from the image is explained in Section 3.4.

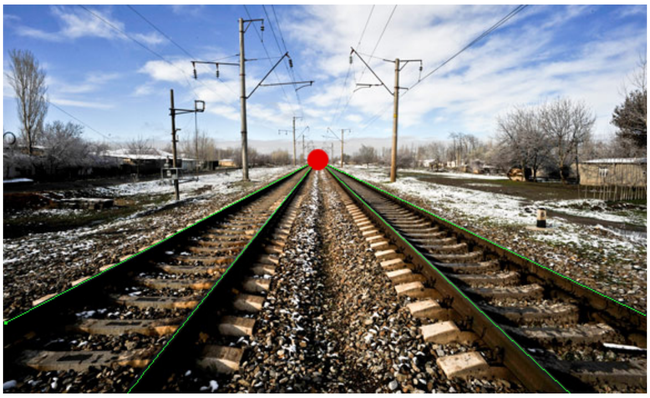


Figure 3.3: Parallel lines converging in a vanishing point.

A vanishing point V_x is defined at infinity, in homogeneous 3D coordinates, as $[1, 0, 0, 0]^T$. Applied to Equation (3.3) with the CCS aligned to the WCS ($T = 0$) it is possible to obtain useful relationships to find the value of the searched variables:

$$\left\{ \begin{array}{lcl} \lambda u_{v_x} & = & fR_{11} + u_0 R_{31} \\ \lambda v_{v_x} & = & fR_{21} + v_0 R_{31} \\ \lambda & = & R_{31} \end{array} \right. \tag{3.5}$$

And substituting the value of λ:

$$\begin{cases} u_{v_x} & = & f\dfrac{R_{11}}{R_{31}} + u_0 \\ v_{v_x} & = & f\dfrac{R_{21}}{R_{31}} + v_0 \end{cases} \tag{3.6}$$

In a similar way a vanishing point V_y is defined at infinity, in homogeneous 3D coordinates, as $[0, 1, 0, 0]^T$. Following the same previous steps an analogous equation is obtained:

$$\begin{cases} u_{v_y} & = & f\dfrac{R_{12}}{R_{32}} + u_0 \\ v_{v_y} & = & f\dfrac{R_{22}}{R_{32}} + v_0 \end{cases} \tag{3.7}$$

There are four unknown variables $(\alpha, \beta, \gamma, f)$, so four expressions are needed. Combining Equations (3.4), (3.6) and (3.7) the necessary expressions are obtained:

$$\begin{cases} u_{v_x} & = & f\dfrac{\cos\gamma\cot\alpha}{\cos\beta} + f\sin\gamma\tan\beta + u_0 \\ v_{v_x} & = & -f\dfrac{\sin\gamma\cot\alpha}{\cos\beta} + f\cos\gamma\tan\beta + v_0 \\ u_{v_y} & = & -f\sin\gamma\cot\beta + u_0 \\ v_{v_y} & = & -f\cos\gamma\cot\beta + v_0 \end{cases} \tag{3.8}$$

The variable isolation is not a complicated task but a little bit laborious. Hence, for the sake of clarity it is summarized into the final expressions:

$$roll = \gamma = \tan^{-1}\left(\frac{u_{v_y} - u_0}{v_{v_y} - v_0}\right) \tag{3.9}$$

$$f = \sqrt{(\sin\gamma(u_{v_x} - u_0) + \cos\gamma(v_{v_x} - v_0))(\sin\gamma(u_0 - u_{v_y}) + \cos\gamma(v_0 - v_{v_y}))} \tag{3.10}$$

$$pitch = \beta = tan^{-1}\left(-\frac{f\sin\gamma}{u_{v_y} - u_0}\right) \tag{3.11}$$

$$yaw = \alpha = tan^{-1}\left(\frac{f\cos\gamma}{(u_{v_x} - u_0)\cos\beta - f\sin\gamma\sin\beta}\right) \tag{3.12}$$

Although in theory the sign of the term under square root in Equation (3.10) should be always positive, it can be negative in practice. That is a good indicator of a wrong vanishing point estimation, to repeat the extraction process.

After explaining how to calibrate a camera from three orthogonal vanishing points, the next paragraphs complete the first necessary steps to do the whole process.

3.3.1 Line extraction

Three sets of orthogonal parallel lines from the image are needed to get the orthogonal vanishing points. The different situations studied in this thesis are depicted in Section 3.4 with an hierarchical tree and a description of every single case. Before that description, it is important to explain how the lines are extracted from the segmented elements.

The first step of the line detection stage is the computation of image derivatives using Sobel edge detector. The line fitting algorithm follows an approach similar to the one suggested by [40]. The gradient direction is quantized into a set of k ranges, where all the edge pixels having an orientation within the certain range fall into the corresponding bin and are assigned to a particular label. All the edge pixels with the same label are then grouped together using connected components algorithm. The line segment candidates are obtained by fitting a line parametrized by an angle θ and distance from the origin $\rho = x\cos\theta + y\sin\theta$.

Each connected component consists of a list of edge pixels (x_i, y_i) with similar gradient orientation, which form the line support regions. The line parameters are directly determined from the eigenvalues λ_1 and λ_2 and eigenvectors v_1 and v_2 of the matrix D associated with the line support region:

$$D = \begin{bmatrix} \sum_i \widetilde{x}_i^2 & \sum_i \widetilde{x}_i\widetilde{y}_i \\ \sum_i \widetilde{x}_i\widetilde{y}_i & \sum_i \widetilde{y}_i^2 \end{bmatrix} \tag{3.13}$$

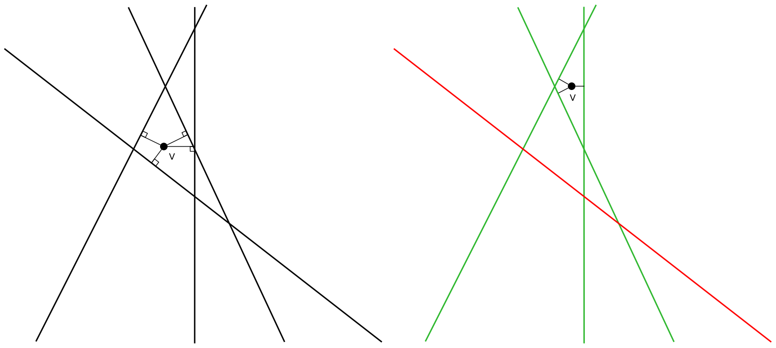
where $\widetilde{x}_i = x_i - \overline{x}$ and $\widetilde{y}_i = y_i - \overline{y}$ are the mean corrected pixels coordinates belonging to a particular connected component and $\overline{x} = \frac{1}{n}\sum_i \widetilde{x}_i$ and $\overline{y} = \frac{1}{n}\sum_i \widetilde{y}_i$. In case of an ideal line, one of the eigenvalues should be zero. The quality of the line fit is characterized by the ratio of the two eigenvalues of matrix D, $v = \frac{\lambda_1}{\lambda_2}$. The line parameters (ρ, θ) are determined from the eigenvectors v_1, v_2, where v_1 is the eigenvector associated with the largest eigenvalue. The line parameters are finally computed as:

$$\begin{aligned} \theta &= atan2(v_1(2), v_1(1)) \\ \rho &= \overline{x}\cos\theta + \overline{y}\sin\theta \end{aligned} \tag{3.14}$$

3.3.2 Vanishing point estimation

Due to noise, a set of imaged parallel lines will generally not intersect in a single point. The “intersection point” can be estimated by determining the point that minimizes the sum of squared perpendicular distances to the fitted lines. However this process is very sensitive to outliers (misclassified segments) so the result can differ considerably from the ideal one, and a previous step is then necessary. To solve this problem, a RANSAC-based algorithm (RANdom Sample Consensus [41]) is used to search for concurrent lines among the detected line segments. Figure 3.4 shows an illustrative example with the vanishing points computed with and without RANSAC processing.

In brief, RANSAC is an algorithm which simultaneously fits parameters and rejects outliers. The idea is that by fitting the parameters to a subset of data consisting of inliers, it is possible to suppress the outliers by rejecting the data which is not consistent with the fitted model. The data which is consistent with this model is called the support of the model. To obtain a subset of inliers in the first place, the whole dataset is randomly



(a) (b)

Figure 3.4: Vanishing point computation from line segments. (a)Wrong vanishing point due to a misclassified line segment. (b) Same solution after RANSAC algorithm. Green segments are inliers and the red segment is an outlier.

sampled and for each sample the parameters are estimated, thus producing a whole population of fitted models. The model with the greatest support is able to be the salient grouping. In this case the model consists of a point, and a sample is performed by choosing randomly pairs of lines. For each pair, their intersection is computed to get putative vanishing points and then the support for this vanishing point is found. An image edge e is able to support the (vanishing) point V_i if there exists a line l through V_i for which the *rms* fitting error to the points on e is less than a threshold (In this work the average distance of the lines to the global least squares solution). Thus the thresholding is done where the measurement errors occur, in the image.

After RANSAC, the point that minimizes the sum of squared perpendicular distances to the fitted lines is obtained. This step is done for each set of lines because the method proposed looks for defined elements in the scene that are known orthogonal (e.g. pedestrians and crosswalks or perpendicular vehicle motion, etc.). In an opposite case where it is known that most of lines are orthogonal (e.g. architectural environments), the algorithm can be launched globally to look for the three most common orientations, with a simultaneous grouping and estimation of vanishing directions using expectation maximization (EM) algorithm [40]. However, such favourable conditions are not always available in traffic scenarios.

3.3.3 Summarized process and preliminary results

The diagram depicted by Figure 3.5 summarizes the proposed camera calibration process. In the following lines, a brief description of each step is presented.

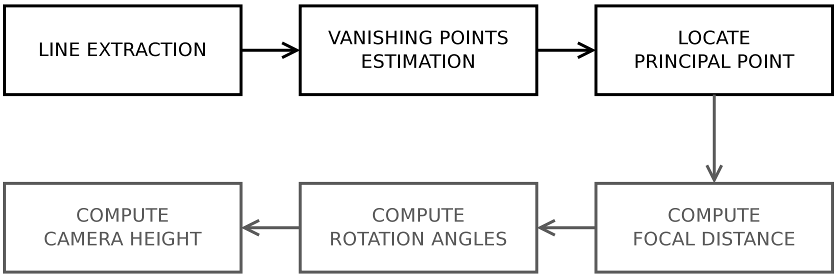
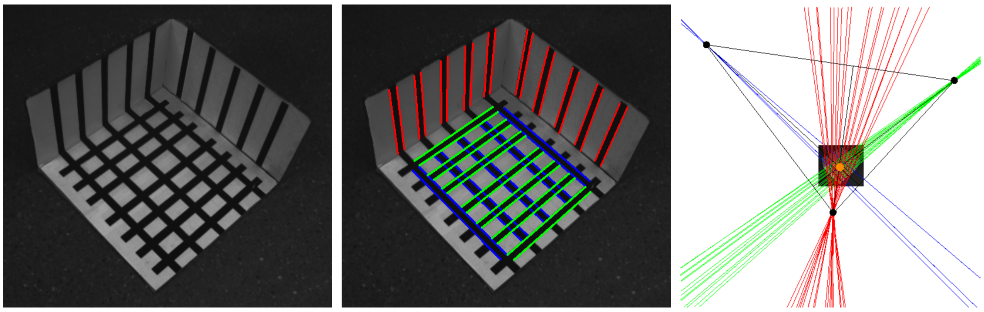


Figure 3.5: Camera auto-calibration process.

- **Line extraction:** extraction of three orthogonal sets of parallel lines.
- **Vanishing points estimation:** the common image intersection points are estimated. Outliers are discarded by RANSAC and the point which minimizes the sum of orthogonal distances to the lines is taken.
- **Locate the principal point:** under the previously described assumption, the orthocenter of the triangle formed by the three orthogonal vanishing points as vertices is the principal point (u_0, v_0).
- **Compute focal length and rotation angles:** the focal length and extrinsic angles (roll, pitch and yaw) are computed by the equations (3.9)-(3.12).
- **Obtain camera height:** if any metric of the scene is available, the camera height (H) can be computed. Otherwise this parameter has to be given or estimated manually in a straightforward step. However, although intrinsic and extrinsic parameters are unknown, in the case of traffic cameras the height is usually associated to the post or building where they are placed. Therefore this parameter is not considered as crucial as the ones explained above.

The first steps are tested in an experiment with a 3D pattern used to simulate three sets of orthogonal lines. As can be seen in Figure 3.6 the orthogonal vanishing points are correctly extracted and the principal point is obtained from them.



(a) (b) (c)

Figure 3.6: Graphic result of the method through a 3D pattern. (a) Original image. (b) Extracted lines. (c) Computed vanishing points and principal point.

After checking the vanishing point extraction, the result of the algorithm is compared with a ground-truth calibration to evaluate the performance of the developed method. The intrinsic parameters are computed by the Matlab Calibration Toolbox [42], and the extrinsic ones by a tripod which provides the angles of the camera with respect to the ground plane. For the Matlab calibration, 15 images have been taken to compute the intrinsic camera parameters. Figure 3.7 shows some examples of the images used.

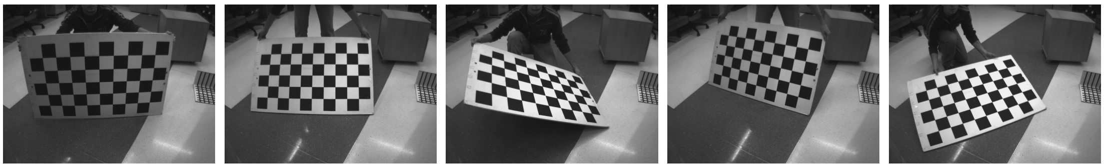


Figure 3.7: Samples of the set of images used to calibrate the camera with the Matlab Calibration Toolbox.

The experiment consist of 15 images, which resolution is 640 × 480, taken in a controlled scenario. Some samples are depicted in Figure 3.8 where the scene remains unaltered but the position and orientation of the camera changes randomly. Figure 3.9 shows the graphic result of one calibration example and the Root Mean Square Error (RMSE) obtained for each parameter is presented in Table 3.1.

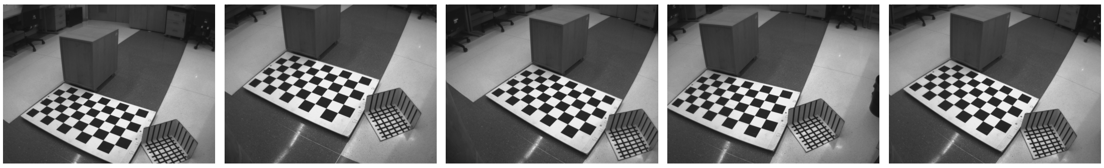
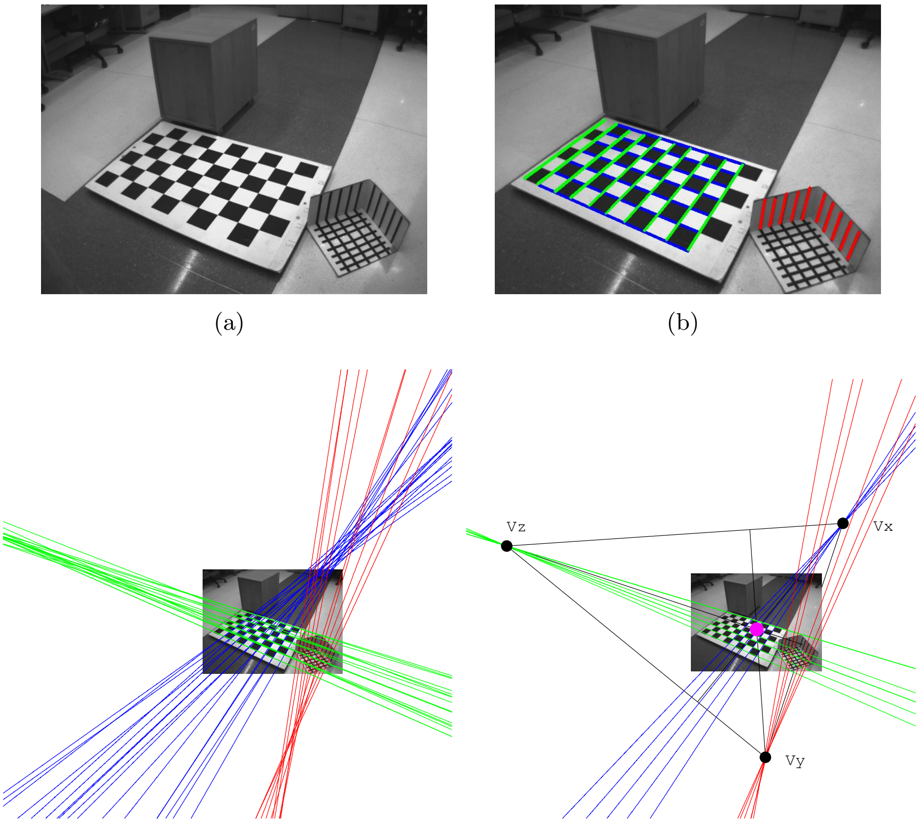


Figure 3.8: Samples of the set of images used for the auto-calibration.



(a)

(b)

(c)

(d)

Figure 3.9: Graphic result of a calibration. (a) Original image. (b) Extracted lines. (c) Initial vanishing point estimation. (d) Lines, vanishing points and orthocenter after RANSAC.

Parameter	Num. Images	ROLL	PITCH	YAW	FOCAL	OP. CENTER
RMSE	15	1.22°	0.71°	1.79°	14.33 pix	7.20 pix

Table 3.1: RMSE between groundtruth calibration and the proposed method.

The error of the extrinsic parameters is less than 2°. Moreover, the focal distance provided by Matlab is 580 pixels, so the deviation error is 2.47%, and the principal point is located in $(u_0, v_0) = (325, 224)$ which means an error of 1.82%. These values are considered more than acceptable for the proposed system.

3.4 Hierarchical auto-calibration system based on the scene

Once the procedure to calibrate a camera from three orthogonal vanishing points has been described, this section presents the proposed process to extract these points from the image. Depending on which elements appear in the scene and the chance of using camera zoom, 5 levels have been established to determine the hierarchy of each developed method and the priority of the solution adopted. The system will choose the available option with higher hierarchy level, corresponding to the smallest number. Levels 1, 2 and 3 provide complete automatic calibration; level 4 needs some assumptions or inputs; and level 5 means manual calibration.

3.4.1 Vanishing point extraction options

Before presenting the hierarchical tree, and to make its comprehension easier, the different options developed in the thesis to obtain the vanishing points and optical center are summarized. In next subsections they are widely explained.

- **Zoom:** when zooming, if several features of the image are matched between frames they converge in a common point which corresponds to the optical center. If this point is known, only two additional vanishing points are needed to compute the rest of the parameters. Subsection 3.4.3 describes the developed process.

- **Crosswalk:** the alternate white and gray stripes painted on the road surface provide a perfect environment to obtain two perpendicular sets of parallel lines. It means that two vanishing points of the ground plane can be extracted, as explained in Subsection 3.4.4.

- **Pedestrians:** humans are roughly vertical while they stand or walk. This characteristic makes them very useful to extract perpendicular lines to the ground. In the case of a structured scene, it is also possible to extract lines of the elements parallel to them (see Subsection 3.4.5).

- **Vehicle motion:** if one vanishing point from the ground plane is needed, it can be obtained from vehicles moving along the main motion direction. Moreover, if there is a perpendicular intersection (in 3D coordinates) in the scene, vehicles moving along the two main directions will provide perpendicular sets of parallel lines corresponding to the two ground plane vanishing points. This procedure is described in Subsection 3.4.6.

- **Structured scene:** in the case of scenes exhibiting a considerable number of orthogonal architectural elements, a last option is available (although less common and effective) to extract the three orthogonal vanishing points by brute force gradient analysis. The extraction process is explained in subsection 3.4.8.

- **Optical center assumption:** there are some cases in which it is not possible to obtain one of the three vanishing points. One option is to enter manually one set of parallel lines, but the autonomy of the system is reduced. To solve this problem, it is possible to assume that the optical center is located on the center of the image, although a small error is committed. Subsection 3.4.7 analyses its advantages and drawbacks.

3.4.2 Hierarchical tree

Figure 3.10 depicts the hierarchical tree proposed in a flowchart. For the sake of clarity, each level has been separated by different lines and colors. As explained before, top levels are the preferred cases, so in case of multiple options available, the system will choose the higher ones.

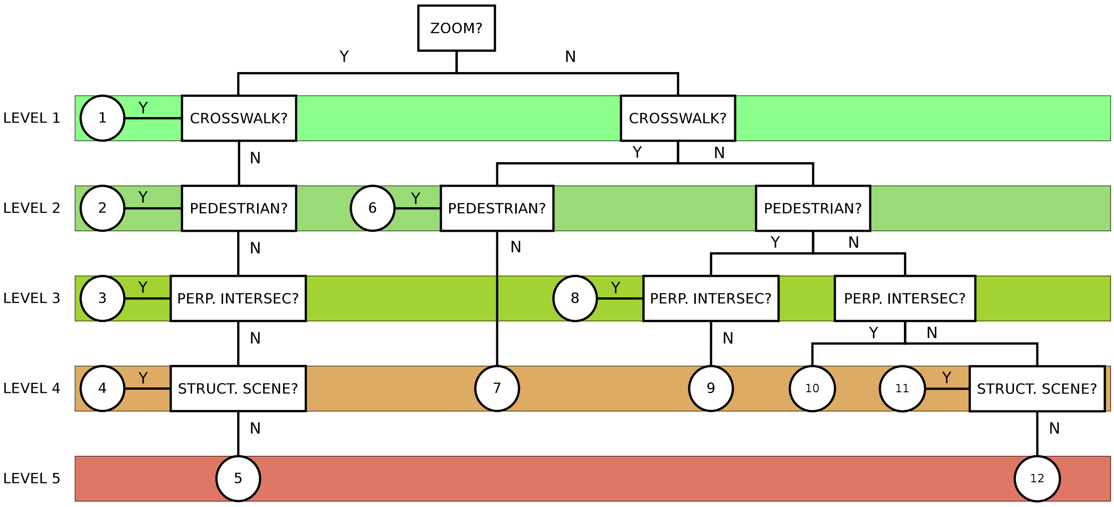


Figure 3.10: Hierarchical calibration tree used. Note: *Perp. Intersec* means perpendicular intersection; and *Struct. Scene* means structured scene.

The different possible cases are described as follows:

- **Case 1:** principal point through zooming process and the two ground plane vanishing points from a crosswalk.
- **Case 2:** principal point through zooming process, the vertical vanishing point from pedestrians and one vanishing point of the ground plane obtained by vehicles moving along the main motion direction.
- **Case 3:** principal point through zooming process and the two ground plane vanishing points from the vehicles moving along the two main directions of the perpendicular intersection.
- **Case 4:** principal point through zooming process and two vanishing points due to parallel lines of the structured scene.
- **Case 5:** principal point through zooming process and manual input of two set of parallel lines, due to absence of necessary information of the scene.
- **Case 6:** the two ground plane vanishing points from a crosswalk and the vertical vanishing point from pedestrians.
- **Case 7:** the two ground plane vanishing points from a crosswalk and either the principal point assumed as the center of the image or manual input of *vertical* lines.
- **Case 8:** the vertical vanishing point from pedestrians and the two ground plane vanishing points from the vehicles moving along the two main directions of the perpendicular intersection.

- **Case 9:** the vertical vanishing point from pedestrians, one vanishing point of the ground plane obtained by vehicles moving along the main motion direction and the principal point assumed as the center of the image. Otherwise manual input of two sets of lines of the ground plane.

- **Case 10:** the two ground plane vanishing points from the vehicles moving along the two main directions of the perpendicular intersection and either the principal point assumed as the center of the image or manual input of *vertical* lines.

- **Case 11:** three vanishing points from the main orthogonal lines of the scene.

- **Case 12:** manual input of three set of orthogonal lines due to total absence of necessary information in the scene.

3.4.3 Principal point through camera zoom

The objective is to find three orthogonal vanishing points and compute the principal point through them. However, if the equations are analysed, after this step only two points are required. Therefore, if it is possible to find the principal point, only two additional vanishing points will be necessary.

When zooming, if several features of the image are matched between frames, the lines which join the previous and new feature positions converge in a common point which corresponds with the optical center. To demonstrate this phenomenon the situation of Figure 3.11 is outlined.

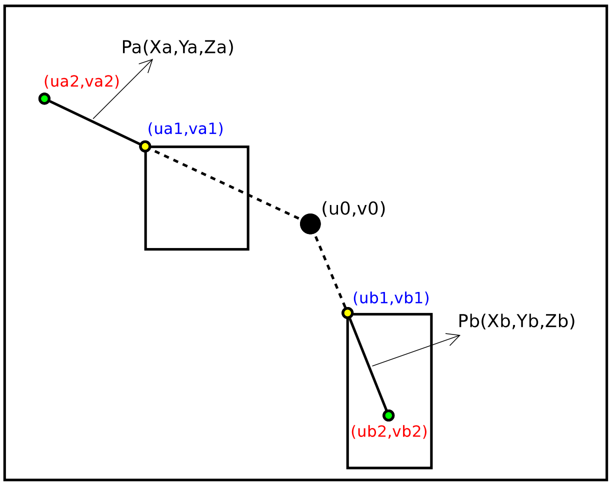


Figure 3.11: Situation to analyse the relation between zoom and optical flow.

The objective is to find if the segments which join (u_{a2}, v_{a2}) to (u_{a1}, v_{a1}) and (u_{b2}, v_{b2}) to (u_{b1}, v_{b1}) have a common point corresponding to the optical center. For this purpose it is necessary to use again the pin-hole camera model of the Figure 3.1 to obtain a geometric relationship between the 3D point, which does not change with zoom, and the point in the image which change with the focal length $(f1 \rightarrow f2)$:

$$\begin{cases} u &= f\frac{X}{Z} + u_0 \\ v &= f\frac{Y}{Z} + v_0 \end{cases} \tag{3.15}$$

With simple geometric line analysis it is known that the lines which pass through (u_{a1}, v_{a1}) and (u_{b1}, v_{b1}) are:

$$\begin{cases} v - v_{a_1} &= m_a(u - u_{a_1}) \\ v - v_{b_1} &= m_b(u - u_{b_1}) \end{cases} \tag{3.16}$$

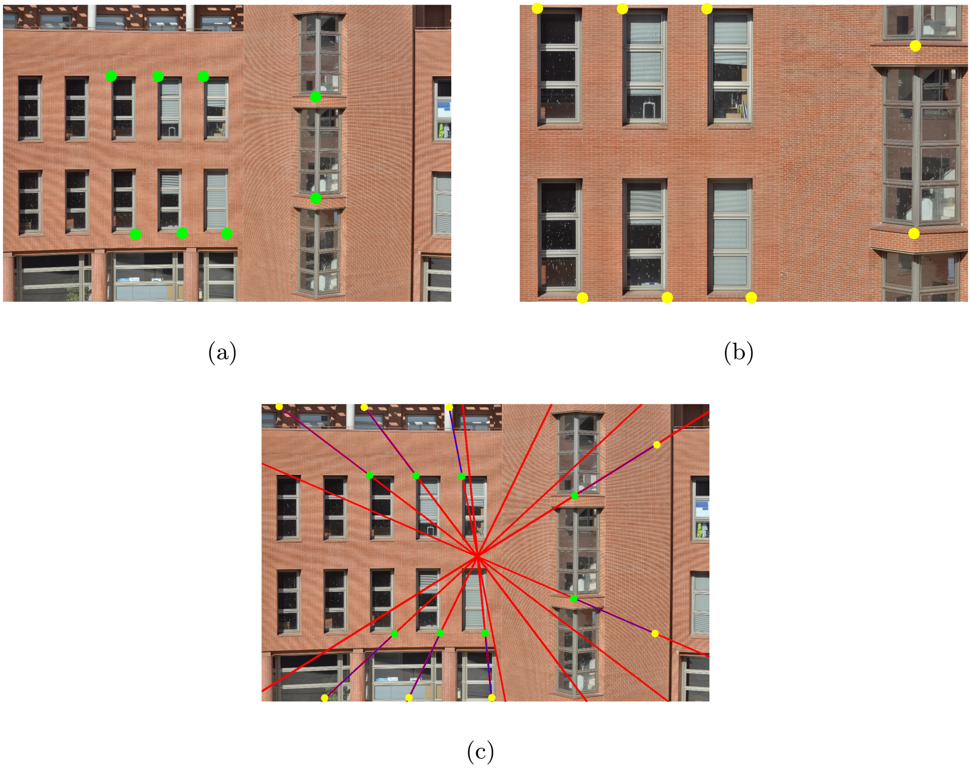
where m_i is the slope of the lines with the form:

$$\begin{cases} m_a = \dfrac{v_{a_2} - v_{a_1}}{u_{a_2} - u_{a_1}} = \dfrac{(f_2\frac{Y_a}{Z_a} + v_0) - (f_1\frac{Y_a}{Z_a} + v_0)}{(f_2\frac{X_a}{Z_a} + u_0) - (f_1\frac{X_a}{Z_a} + u_0)} = \dfrac{Y_a}{X_a} \\ m_b = \dfrac{v_{b_2} - v_{b_1}}{u_{b_2} - u_{b_1}} = \dfrac{(f_2\frac{Y_b}{Z_b} + v_0) - (f_1\frac{Y_b}{Z_b} + v_0)}{(f_2\frac{X_b}{Z_b} + u_0) - (f_1\frac{X_b}{Z_b} + u_0)} = \dfrac{Y_b}{X_b} \end{cases} \tag{3.17}$$

Therefore isolating a point (u, v), the following expression is derived:

$$v = \frac{Y_a}{X_a}(u - u_0) + v_0 \tag{3.18}$$

And finally if $u = u_0 \rightarrow v = v_0$, which is the result searched. Figure 3.12 shows an example of this phenomenon: An image was taken before and after zooming and the matched features converge to the same point, the optical center.


(a)

(b)

(c)

Figure 3.12: Principal point computation through camera zoom. (a) Image before zooming and features extracted. (b) Image after zooming and features extracted. (c) Feature matching. The common point corresponds to the optical center.

3.4.4 Zebra crossing vanishing point extraction

A common intersection scenario usually has zebra crossings like the one presented in Figure 3.13.

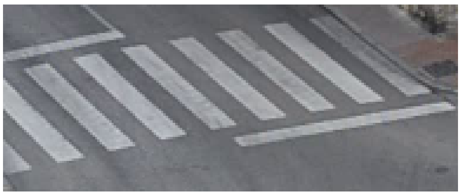


Figure 3.13: Example of zebra crossing.

The alternate white and gray stripes, painted on the road surface, provide a perfect environment to obtain two perpendicular sets of parallel lines. It means that the two vanishing points from the ground plane can be obtained.

To detect if there are crosswalks in the image for a posterior analysis, the following steps are done.

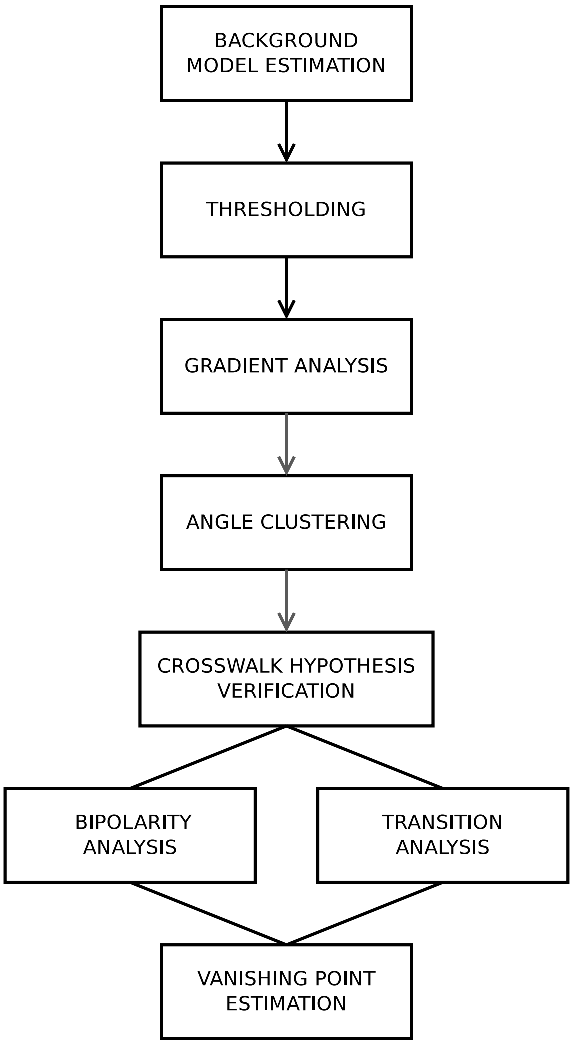

BACKGROUND MODEL ESTIMATION
THRESHOLDING
GRADIENT ANALYSIS
ANGLE CLUSTERING
CROSSWALK HYPOTHESIS VERIFICATION
BIPOLARITY ANALYSIS
TRANSITION ANALYSIS
VANISHING POINT ESTIMATION

Figure 3.14: Crosswalk detection process.

- **Background model estimation:** by the background subtraction algorithm presented in Section 4.2.1, the background model is extracted to look for crosswalk candidates without moving objects that can occlude them, or sudden illumination changes.

- **Thresholding:** as the typical zebra crossing has a strong white component, a thresholding step is done in order to highlight the white stripes.

- **Gradient analysis:** the line extraction algorithm explained in Section 3.3.1 is used in order to obtain the straight lines of the scene, necessary for the vanishing point estimation.

- **Angle clustering:** all the lines extracted are initially grouped by angle in order to distinguish between different kind of candidates. To separate lines with close angles but from different crosswalk candidates a RANSAC filter is applied. The input of the algorithm is the distance from each line to the rest of the cluster. Segments that do not belong to the neighbourhood are included in a different cluster or discarded.

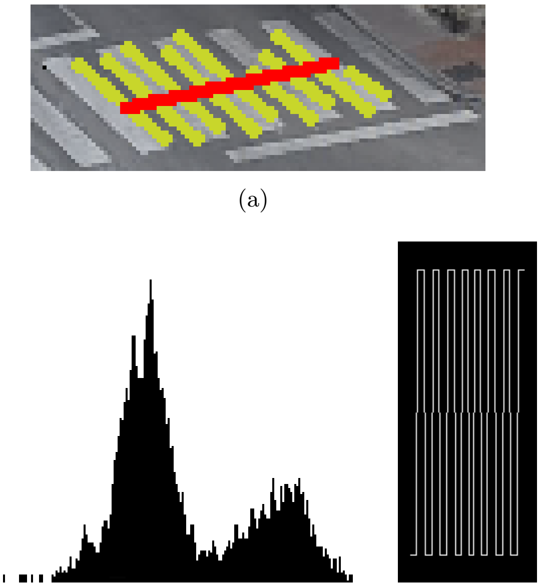
- **Verify crosswalk hypothesis:** a confidence factor of each candidate is taken in order to decide if whether or not it can be consider as a zebra crossing. In the case of more than one valid candidate, the system chooses the one with the highest confidence factor. This factor is based on two indicators:

  1. Bipolarity analysis. A gray color based histogram is constructed to analyse the bipolarity component of a crosswalk. In case of a zebra crossing, this histogram should have two representative gaussian components, as shown in Figure 3.15(b).
  2. Transition analysis. The b/w transitions (in the binary image) are analysed, in order to measure the number of changes and how constant the width of the stripes is. This process is done through a transitions binary pattern constructed by the values of the line which best represents the direction of the crosswalk. This line is obtained fitting by RANSAC the center of the gradient lines extracted for each zebra crossing.

  The corresponding gradients (in yellow), representing line (in red), bipolar histogram and transition pattern of the crosswalk of Figure 3.13 are represented in Figure 3.15.

- **First vanishing point estimation:** The vanishing point corresponding to the main direction of the crosswalk stripes is computed as explained in Section 3.3.2, with the gradients extracted previously.

- **Second vanishing point estimation:** Due to the small size and the irregularity of the perpendicular segments of the stripes, the gradient analysis is not accurate enough to obtain the desired set of parallel lines. To solve this problem, the centroid of each segment is computed as the intersection of the central line of the stripe with the end of the stripe. All the points obtained are fitted to a line by RANSAC and the intersection between the upper and lower lane is consider the second vanishing point. The process is represented in Figure 3.16.



(a)

(b) (c)

Figure 3.15: Confidence factor indicators of a crosswalk. (a) Gradients and fitted representing line. (b) Bipolar histogram. (c) Transitions binary pattern.

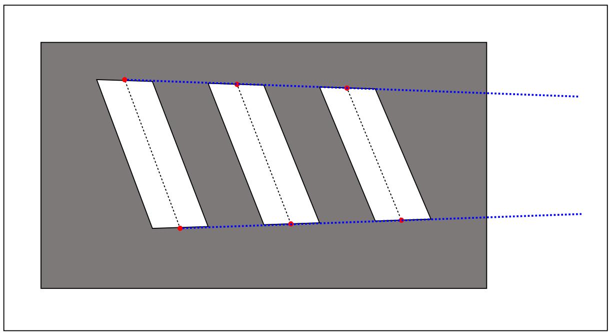
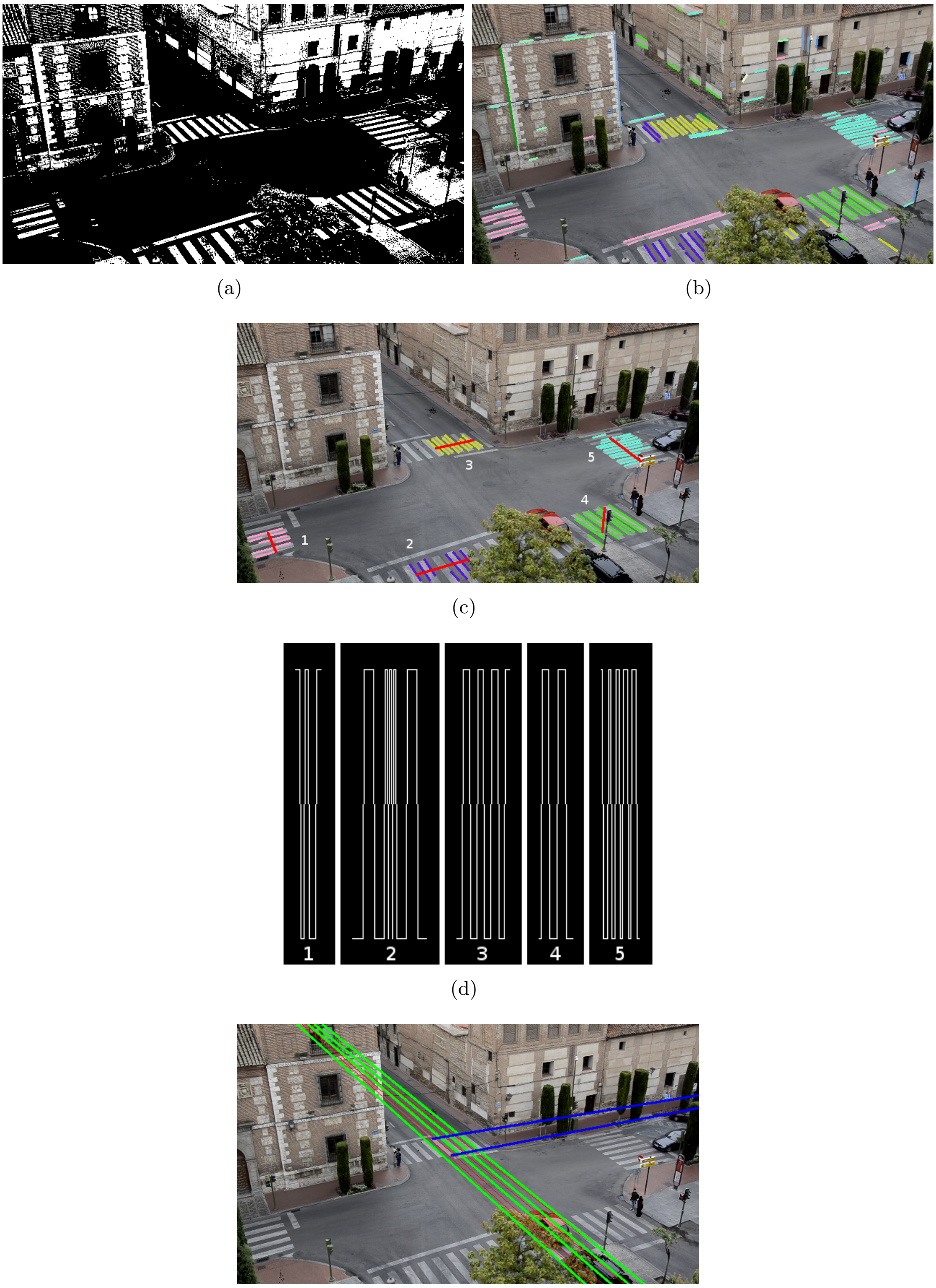


Figure 3.16: Extraction of the second vanishing point from a crosswalk.

An example of the whole method proposed is depicted in the Figure 3.17. Firstly, the background model image is binarized (Figure 3.17(a)), and the lines are extracted by gradient analysis and grouped by angle (Figure 3.17(b)). After that, a RANSAC-based filter is applied to get the final candidates (FIgure 3.17(c)). The red line is the one which best fits the candidate. Bipolarity and transition analysis are then done in order to obtain the confidence factor with the following results:

- Candidate 1 = 0.10 (Low value due to an irregular pattern).
- Candidate 2 = 0.14 (Low value due to a white stripe detected with black holes).
- Candidate 3 = 0.96 (Good pattern. This is the candidate chosen).
- Candidate 4 = 0.40 (Bad result due to the interruption of the traffic light).
- Candidate 5 = 0.77 (Acceptable value, but more irregular than candidate 3).

Finally, the vanishing points are computed (Figure 3.17(e)) with the following results: $V_x = (-212.64, -266.07)$ and $V_z = (950.23, 59.11)$



(a)

(b)

(c)

(d)

(e)

Figure 3.17: Crosswalk detection example. (a) Binarized background model. (b) Line extraction. (c) Grouped candidates with testing lines in red. (d) Transition pattern of candidates 1 to 5. (e) Parallel lines to compute the vanishing points.

3.4.5 Pedestrian vanishing point extraction

Humans are roughly vertical while they stand or walk. This property makes them very useful to get perpendicular lines to the ground, to compute the vertical vanishing point. One option is to extract the vertical component of each pedestrian to form the necessary set of parallel lines, as done by Hodlmoser et al. [11]. However, the cameras in common traffic scenarios are usually located quite higher than the situations proposed by the authors in the paper, and small pedestrians can derive into erroneous lines extractions. Traffic scenes provide a lot of structured elements with vertical components (walls, lampposts, traffic lights, etc.), that can be used to increase the performance and quality of the system. The developed algorithm is based on this idea, and it is divided into the following steps: pedestrian detection with vertical component extraction, scene analysis and vanishing point computation.

The aim of this method is to detect pedestrians with no false positives, to avoid lines that are not perpendicular to the ground. Therefore, it is not crucial to detect all the pedestrians in the image but it is important to be sure that the detected objects are humans. In order to obtain useful candidates for vertical lines extraction two kinds of parameters for every moving object are obtained: the motion direction and the main axis direction. The difference of these directions is quite significant for moving pedestrians while it is very small for vehicles, in most cases of camera view as shown in Figure 3.18. As a result, this parameter is taken as a discriminant feature for coarse classification along blob aspect ratio constraints like $height > 3 \cdot width$. It is supposed that cast shadows have been previously deleted as described in Section 4.2.3.

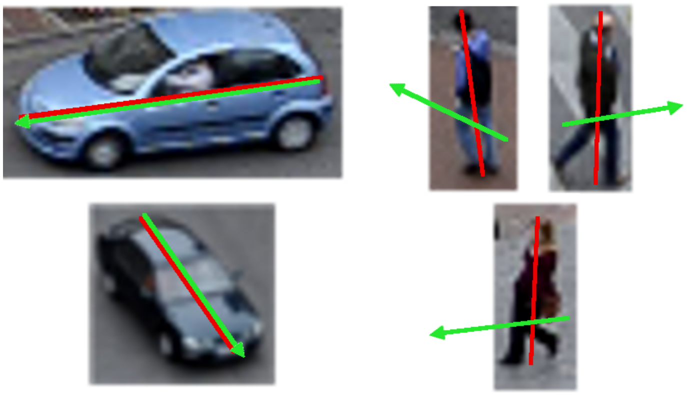


Figure 3.18: Angles used to differentiate between vehicles and pedestrian. Green arrowhead stands for velocity direction; Red line stands for main axis direction.

It is evident that this classification is not very accurate, but in practice it is good enough to get valid pedestrians useful to extract vertical lines.

To get the motion direction of the blob, the average motion of the blob features is taken. This feature analysis is explained in the next chapter. In the case of the main axis direction of the blob θ, three different approaches have been used: moment analysis, principal component analysis and RANSAC estimation. The direction estimated by moment analysis is defined as:

$$\theta_{moment} = \tan^{-1}(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}) \tag{3.19}$$

where μ_{pq} is the central moment of order (p, q).

Principal component analysis (PCA) is equivalent to major axis regressions, so the largest axis can be considered as the vertical component. And finally RANSAC algorithm takes the centroid of each candidate row to estimate the line that corresponds to the main axis of the pedestrian. When these three methods obtain similar results and the blob aspect ratio is valid, the candidate is considered a pedestrian. Figure 3.19 shows an example of vertical line extraction.

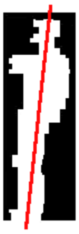


Figure 3.19: Pedestrian main axis extraction.

At the same time, a gradient line extraction of the image is done in order to extract all the possible structured elements. The angle of the vertical components of the pedestrians will be compared to the lines extracted and, in case of matching, the lines will be saved to compute afterwards the vanishing point. Due to the perspective of the camera, a perpendicular line to the ground in the image has different angles depending on the position. Moreover, because of the negative *pitch* the vertical vanishing point has to be positive. Therefore the image is divided into five quadrants following the angle constraints depicted in Figure 3.20.

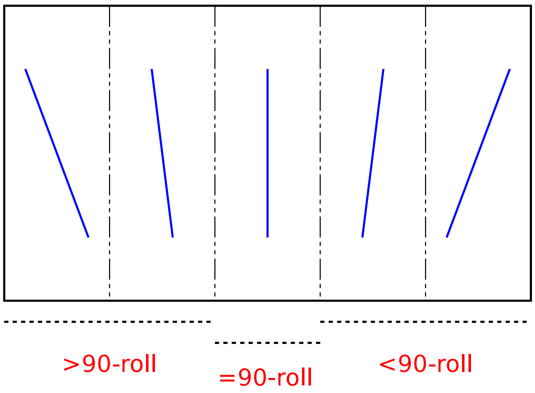
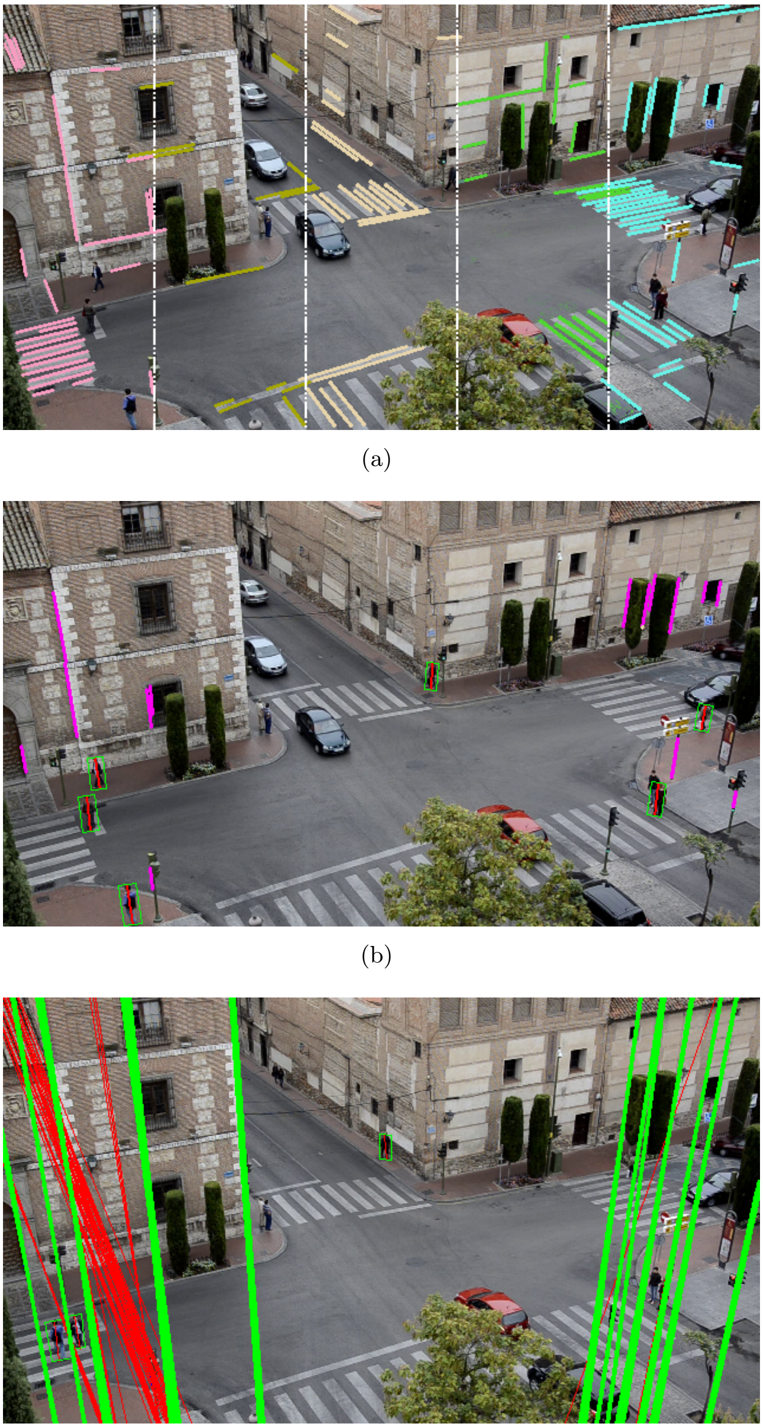


Figure 3.20: Quadrants and angle constraints due to perspective.

Figure 3.21 depicts an example of the developed method. Figure 3.21(a) represents the lines extracted from the scene, with different colors depending on the belonging quadrant. Figure 3.21(b) shows the detected pedestrian inside a green box with the estimated vertical component in red, and the matched vertical lines in cyan. Finally, Figure 3.21(c) depicts the estimation of the vertical vanishing point with all the accumulated vertical lines. Red lines are the outliers and green lines the inliers for the RANSAC-based method explained before. The resulting vanishing point is: $V_y = (305.52, 1698.65)$.



(a)

(b)

(c)

Figure 3.21: Vertical vanishing point extraction example. (a) Extracted scene lines divided by 5 quadrants. (b) Detected pedestrians with red vertical component and vertical matches in cyan. (c) RANSAC vanishing point estimation with red outliers and green inliers.

3.4.6 Vehicle motion vanishing point extraction

One of the properties of the traffic scenarios is that many vehicles drive in the same or inverse direction of the 3D world. Therefore the main axis of these vehicles are parallel to each other, and also parallel to the ground plane. This supplies important information to extract horizontal vanishing points.

As explained in the hierarchical calibration tree (Figure 3.10), there are cases that need only one ground plane vanishing point while others need two. In case of computing the optical center (either by zooming analysis or assuming it as the image center) and detecting pedestrians, only one vanishing point from the ground plane is needed, in any direction. On other hand, in case of needing two ground plane vanishing points and if a perpendicular intersection (in 3D coordinates) is present in the scene, vehicles moving along the two main directions will provide perpendicular sets of parallel lines corresponding to the two ground plane vanishing points. In both cases the followed process is similar, done either for one direction or two.

Firstly, the main motion directions are extracted. For this purpose, a feature optical flow analysis of the foreground blobs is done and their motion direction is saved into an histogram. Once it is constructed after a determined number of frames, an EM algorithm is used to fit the histograms into gaussians in order to get the principal components of the movement. Figure 3.22 shows an example of a perpendicular intersection, where the features of the foreground objects are tracked by optical flow and the motion direction histogram with the gaussian components in red is computed. The vertical axis corresponds to the frequency of the angle, and the horizontal axis corresponds to the angle value between 0° and 180°. In this case, the extracted main directions are 14° and 137°. These values are not perpendicular in image coordinates due to the perspective projection.

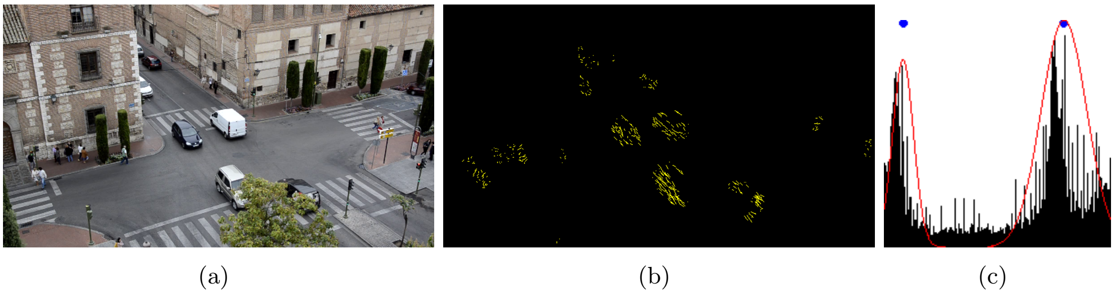


Figure 3.22: Example of main motion directions extracted in a perpendicular intersection. (a) Perpendicular intersection. (b) Foreground optical flow analysis. (c) Histogram of directions and fitted gaussians in red.

After getting the main directions of the scene, the motion of each foreground blob is analysed. In case of detecting motion in the computed directions, the gradients of the blob are extracted (see Section 3.3.1) in order to look for parallel lines with the mentioned angles. Once obtaining a representative number of parallel segments, the RANSAC method proposed in Section 3.3.2 is used again. Finally, the searched vanishing points are obtained. Figure 3.23 shows an example of two ground plane vanishing point extraction using the method explained in this section.

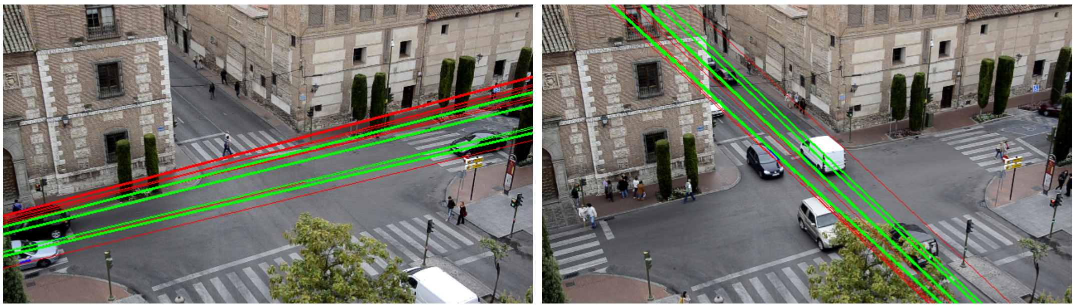


Figure 3.23: Example of ground plane vanishing point extraction in a perpendicular intersection.

This method has to manage the drawback of that not all vehicles have exactly the same trajectory and the same gradients along the main directions. Section 3.6 analyses the maximum vanishing point error that can be assumed.

3.4.7 Optical center assumption

There are some cases in which the automatic extraction of one vanishing point is not available. To manage this problem, one option is to manually enter a set of parallel lines, but the autonomy of the system is reduced. Other possibility is to assume that the optical center is on the center of the image, although a small error is committed depending on the camera and the lens. Figure 3.24 represents an example where two vanishing points are known and the third one is obtained from these points and the center of the image. The first vanishing point is extracted from the crosswalk and the number 2 from pedestrians. The green line is formed by joining the optical center with the mathematically isolated third vanishing point to demonstrate it is orthogonal to the others. In the results chapter, the optical center assumption is tested with the rest of methods to show the committed error.

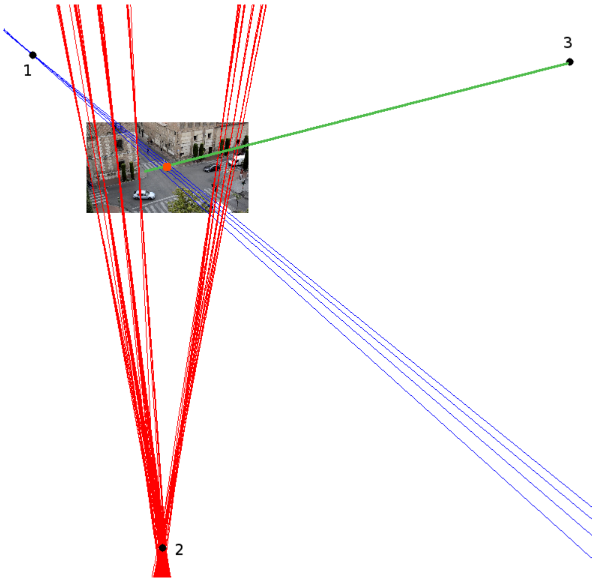


Figure 3.24: Example of optical center assumption.

3.4.8 Structured scenarios vanishing point extraction

In the case of having a considerable number of architectural elements in the scene, a last option for an autonomous calibration is available (although less common and effective). It consist on extracting the vanishing points by brute force gradient analysis, assuming that the three sets of parallel lines with most number of lines are orthogonal. To group the lines, J-Linkage algorithm [43] is used. This method is based on the work of Tardif in [7], although he does not look for orthogonal vanishing points.

The input of the algorithm is a set of N edges, that can be obtained by the method proposed in Section 3.3.1. The output is a set of vanishing points and a classification for each edge: assigned to a vanishing point or marked as an outlier. The solution relies on the J-Linkage algorithm to perform the classification.

The first step is to randomly choose M minimal sample sets of two edges $S_{1...M}$ and to compute a vanishing point hypothesis $v_m = V(S_m, 1)$ for each of them (1 is a vector of ones, i.e. the weights are equal). The second step consists of constructing the preference matrix P, a $N \times M$ Boolean matrix. Each row corresponds to an edge ε_n and each column to a hypothesis v_m. The consensus set of each hypothesis is computed and copied to the m^{th} column of P. An example of matrix P is given in Figure 3.25. Each row r of P is called the characteristic function of the preference set of the edge ε_n: the m^{th} entry is 1 if v_m and ε_n are consistent.

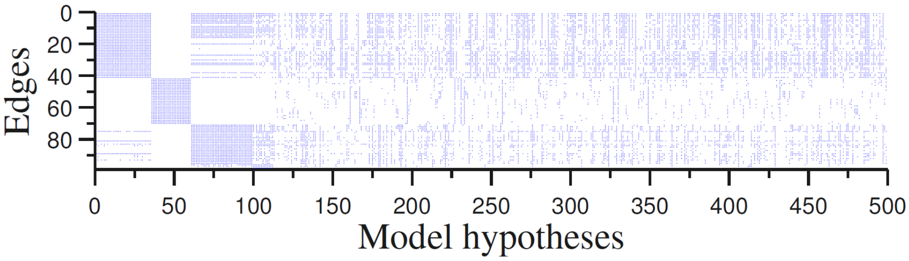


Figure 3.25: Example of Preference matrix for N=100 edges and M=500 vanishing point hypothesis.

The J-Linkage algorithm is based on the assumption that edges corresponding to the same vanishing point tend to have similar preference sets. Indeed, any non-degenerate choice of two edges corresponding to the same vanishing point should yield solutions with similar, if not identical, consensus sets. The algorithm represents the edges by their preference set and clusters them as described below. Note that at this point, the hypothesized vanishing points are completely ignored by the algorithm. The algorithm defines the preference set of a cluster of edges as the intersection of the preference sets of its members. It also uses the Jaccard distance between two clusters, given by:

$$d_j(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \tag{3.20}$$

where A and B are the preference sets of each of them. It equals 0 if the sets are identical and 1 if they are disjoint. The algorithm proceeds by placing each edge in its own cluster. At each iteration, the two clusters with minimal Jaccard distance are merged together. The operation is repeated until the distance between all clusters is equal to 1. Once clusters of edges are formed, a vanishing point can be computed for each of them and refined by the RANSAC-based method used in this thesis.

Figure 3.26 shows the orthogonal lines extracted in a structured scenario to compute the three orthogonal vanishing points.

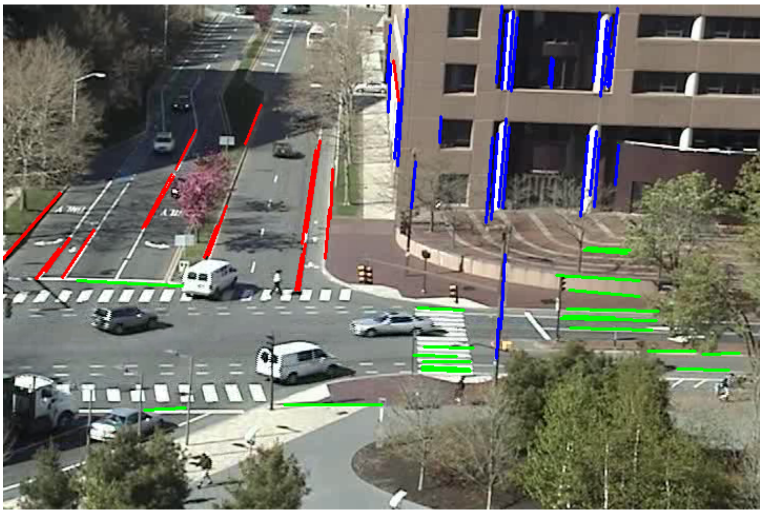


Figure 3.26: Extracted lines in a structured scenario to obtain three orthogonal vanishing points automatically.

3.4.9 Manual vanishing point extraction

In some cases, it is not possible to extract the necessary vanishing points automatically and the user's interaction is needed. For these situations, an interactive tool has been developed to draw the lines to get the orthogonal vanishing points in any cases. The user draws a set of parallel lines and the system computes the intersection with the method explained in Section 3.3.2. It is important to emphasize that this option is still an advantage against other methods, due to the chance to calibrate the camera in a short time and without needing extra information or calibration patterns, in spite of the user interaction.

The examples depicted in Figure 3.6 and 3.9 were taken with the interactive tool. The drawn blue and green lines correspond to the ground plane and set of red lines to the vertical plane.

3.5 Experimental results

In the next section, the results of the auto-calibration stage are presented. There are no public useful databases to compare the developed method with previous works, therefore the calibration based on the manual vanishing point extraction (the method which best fits the vehicles into prisms) is considered the groundtruth of the system.

From the total amount of videos, two different representative scenes have been selected to show the performance of the proposed methods. It is described through two tables, which include all the information obtained from the experiments (Tables 3.2 and 3.3) and a list of conclusions. Finally, a comparative table summarizes the average error of each hierarchical tree case, to demonstrate if the assumed hierarchy is correct.

3.5.1 Auto-calibration results table legend

For the sake of clarity, the information of each row and column of the tables is explained in the following paragraphs:

- **Case:** the different possible cases of the hierarchical tree:
  - Case 1: zoom and crosswalk.
  - Case 2: zoom, pedestrians and main motion direction of vehicles.
  - Case 3: zoom and perpendicular intersection.
  - Case 4: zoom and structured scene.
  - Case 6: crosswalk and pedestrians.
  - Case 7: crosswalk and OC assumption.
  - Case 8: pedestrians and perpendicular intersection.
  - Case 9: pedestrians, main motion direction of vehicles and OC assumption.
  - Case 10: perpendicular intersection and OC assumption.
  - Case 11: structured scene.
  - Cases 5 and 12: manual input of lines (represented as Case 12/5).

  Two more cases have been created after testing all videos, with a different combination of the explained methods, because of an improvement of the results. These new cases are represented with the number of the original method and a subindex (1_2 and 11_2):

  - Case 1_2: takes the principal point through zooming process, one ground plane vanishing point from the main direction of a crosswalk, and the vertical vanishing point from the pedestrians and vertical structures. This option is better than case 1 because the vertical vanishing point extraction is the most reliable of all.
  - Case 11_2: assumes the principal point as the center of the image and computes two vanishing points due to two sets of parallel lines of the structured scene. This option improves the results of case 11 because in many situations is easy to find two orthogonal sets of parallel lines, but not three.

- **VPi:** coordinates of the computed vanishing points for each case. VP_1 and VP_2 are the horizontal vanishing points and VP_3 is the vertical one.

- **OC:** coordinates of the computed or assumed principal point of the image.

- **Focal, pitch and roll:** values of the computed intrinsic and extrinsic camera parameters. Yaw is not considered because its variation does not modify the ground plane and does not have impact into the 3D projection.

- **distA, distB, distC:** 3D depth distance from the camera to three selected points of the image. The distance is computed as explained in Subsection 3.5.2, and compared to the one obtained by Google Maps [44] (in the last row of the table).

- **vol1, vol2, vol3:** volumes of the projected prisms over three different vehicles.

3.5.2 Distance estimation

The distance from the camera to several points of the scene is one of the parameters used to evaluate the performance of the developed method. Due to the loss of depth information, monocular systems need a perspective projection from the 3D world to the image, using the general pin-hole camera model, represented in Figure 3.1 and Equation (3.3).

The focal distance (f), rotation matrix ($R_{3\times3}$) and camera height (H) are known variables, as explained in previous sections. With two image coordinates (u, v), only two expressions are available to isolate the X and Z world coordinates. Therefore, the Y component has to be assumed as 0, and the distances computed from points located in the ground plane.

Therefore the resultant pin-hole equation for a point located on the ground plane has the following form:

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} fR_{11} + u_0 R_{31} & fR_{12} + u_0 R_{32} & fR_{13} + u_0 R_{33} & 0 \\ fR_{21} + v_0 R_{31} & fR_{22} + v_0 R_{32} & fR_{23} + v_0 R_{33} & fH \\ R_{31} & R_{32} & R_{33} & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.21}$$

The final system to isolate X and Z, assuming $Y = 0$ is:

$$\left\{ \begin{array}{rcr} (fR_{11} + (u_0 - u)R_{31})X + (fR_{13} + (u_0 - u)R_{33})Z & = & 0 \\ (fR_{21} + (v_0 - v)R_{31})X + (fR_{23} + (v_0 - v)R_{33})Z & = & -fH \end{array} \right\} \tag{3.22}$$

And the distance is computed from:

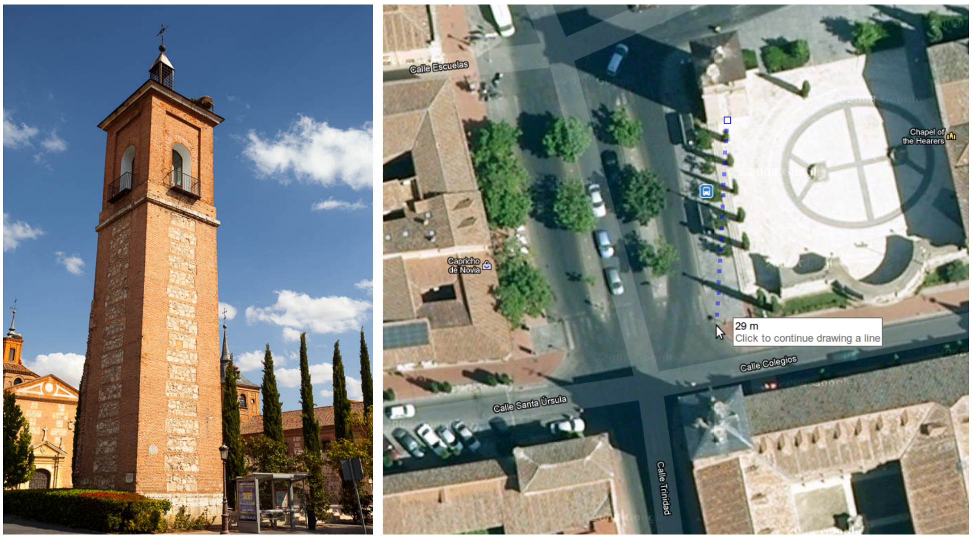
$$distance = \sqrt{Z^2 + X^2} \tag{3.23}$$

The distance computed by these equations is compared with the one obtained by Google Maps. Figure 3.27(b) shows an example of how this distance is extracted from the website.

3.5.3 Experiments

The tests presented in this thesis are performed in two sequences recorded from the top of a tower located in the city center of Alcalá de Henares, and called *Torre de Santa María* (Figure 3.27(a)).

Both sequences provide a zoom change, so zoom-based cases (1 to 4) can be covered. Moreover the scenes have apparently enough structured elements to cover cases 4 and 11. Finally, the first scenario contains an intersection that can be considered perpendicular, therefore cases 3, 8 and 10 are studied.

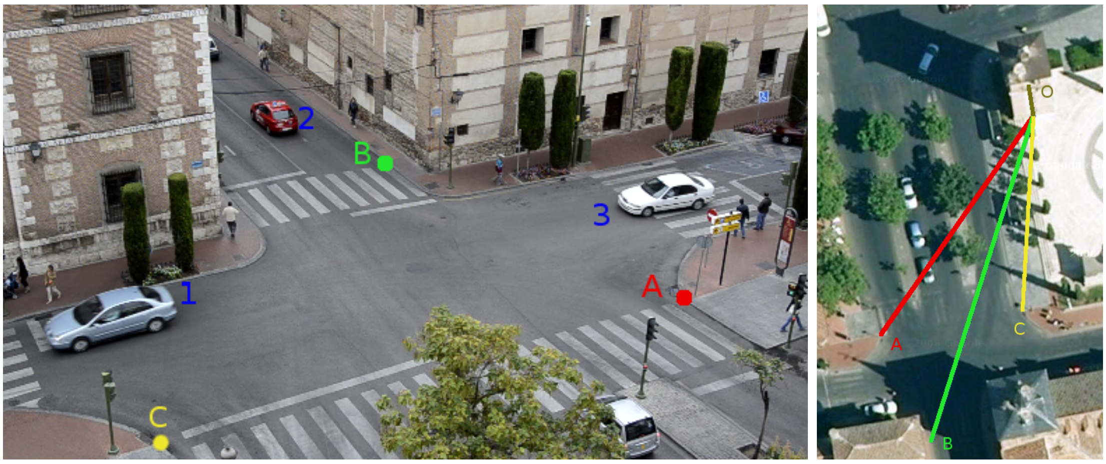


(a) (b)

Figure 3.27: (a) Torre de Santa María, where the camera was located. (b) Example of distance extraction from the tower, with Google Maps.

Test 1

The first scene is presented in Figure 3.28, where three points are selected to measure their distance from the camera. Their values from Google Maps are: $A = 39m$, $B = 50m$ and $C = 29m$. The numbers correspond to the vehicle indexes for the volume of the projected prisms comparison.



(a) (b)

Figure 3.28: Scenario of test 1 and selected points to measure their distance from the camera (located in the tower). (a) Image points and vehicle indexes. (b) Corresponding points from Google Maps.

To analyse the results obtained in this test, Table 3.2 summarizes all the values extracted and computed by the system. Figure 3.29 shows the graphic result of the vehicle prism projection for the manual vanishing point extraction case, and Figure 3.30 depicts the graphic results for the rest of cases.

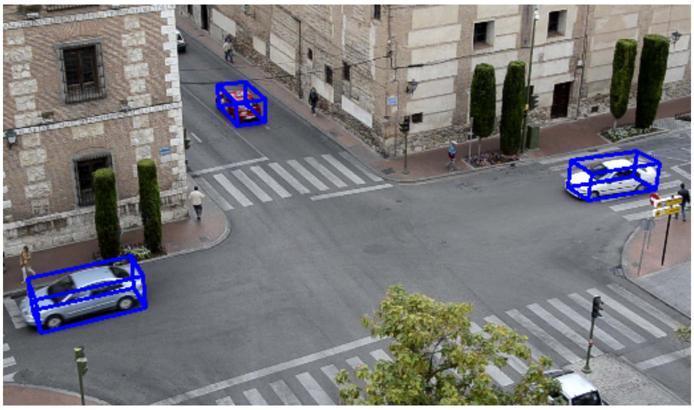


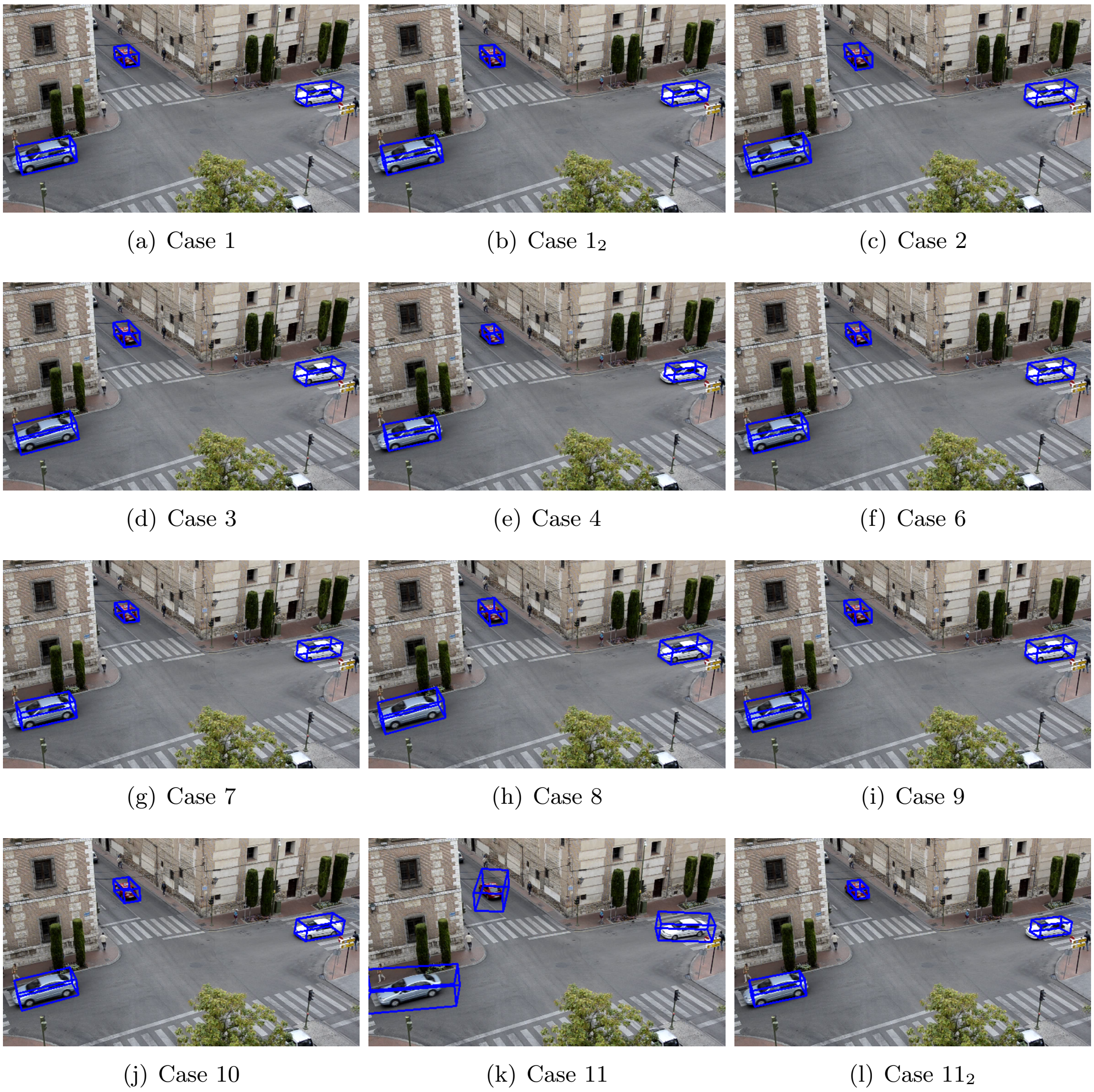
Figure 3.29: Graphic result for the manual vanishing point extraction case.

The partial conclusions after analysing the graphical and numerical results are explained below:

- Due to the strong vertical component of the scene and the presence of pedestrians, the vertical vanishing point is more reliable than the second vanishing point from a crosswalk. Accordingly, although both options are valid, case 1_2 is closer to the groundtruth than case 1.
- Similarly, case 11_2 improves the result of case 11 due to the difficulties to find 3 orthogonal sets of parallel lines automatically. A ground plane vanishing point and the vertical one are extracted correctly, but the second vanishing point computed from the ground plane is not orthogonal. The focal distance of case 1 is pretty similar to the groundtruth one, but the principal point and roll have a considerable error. That situation makes the solution unavailable for this scenario. On the other hand, assuming the principal point as the center of the image and taking only two orthogonal sets of parallel lines, the result becomes more acceptable.
- The assumption of perpendicularity in the intersection is not very accurate. As observed in Figure 3.27(b), the geometry of the scene is more or less perpendicular, but the road lanes, where the vehicles drive, are not exactly perpendicular. Cases 3, 8 and 10 demonstrate deviations in both intrinsic and extrinsic computed parameters. However, it does not have a big impact in the graphical results shown in Figure 3.30.
- Related to the measured distances, it is important to point that the error introduced by Google is unknown, so the groundtruth is an approximation. In this context, the computed results except for the cases of perpendicular intersection are very close to the values provided by the website, which validates the calibration process.
- In terms of 3D prisms projections, only case 11 can be considered erroneous, because of the reasons explained above. As can be seen graphically, the rest of the cases are pretty similar and acceptable.
- In general, the methods that extract the principal point through zooming process are more accurate. The extraction process is very reliable and provides the system strong versatility.

CASE	VP1	VP2	VP3	OC	FOCAL	PITCH	ROLL	distA	distB	distC	vol1	vol2	vol3
12/5	(-123.16,-132.02)	(1550.60,-118.30)	(308.81,1646.98)	**(320.27,180.10)**	**680.35**	**-24.89**	**0.48**	**39.04**	**49.90**	**29.97**	**41271**	**8190**	**22142**
1	(-49.84,-119.79)	(1885.23,-118.01)	(324.17,1789.70)	(325.43,187.64)	700.52	-23.61	0.03	39.79	51.30	30.77	39296	6419	18706
1_2	(-49.84,-119.79)	(1775.60,-103.89)	(311.96,1684.45)	(325.43, 187.64)	681.33	-24.47	0.50	38.77	49.55	29.86	37974	6611	18706
2	(-325.48,-154.28)	(1242.70,-135.73)	(311.96,1684.45)	(325.43,187.64)	713.04	-26.05	0.66	37.22	46.32	29.26	50113	10961	25625
3	(-325.48,-154.28)	(1109.21,-157.55)	(327.39,1332.90)	(325.43,187.64)	623.84	-28.56	-0.12	34.13	42.63	26.84	44581	9153	25018
4	(-10.62,-81.41)	(1640.70,-41.42)	(290.06,1647.07)	(325.43,187.64)	634.46	-23.50	1.37	39.91	51.98	29.67	29817	4091	13098
6	(-49.84,-119.79)	(1885.23,-118.01)	(311.96,1684.45)	(312.53,196.67)	686.69	-24.78	0.06	37.62	47.44	29.15	34015	6652	18461
7	(-49.84,-119.79)	(1885.23,-118.01)	(318.31,1814.20)	(320.00,180.00)	700.64	-23.21	0.06	41.14	53.57	31.48	39296	6818	18706
8	(-325.48,-154.28)	(1109.21,-157.55)	(311.96,1684.45)	(307.72,120.26)	653.30	-22.67	-0.12	50.69	73.59	35.96	52552	10959	28619
9	(-325.48,-154.28)	(1259.40,-145.57)	(311.96,1684.45)	(320.00,180.00)	712.36	-25.35	0.30	38.61	48.72	30.10	48926	10070	27672
10	(-325.48,-154.28)	(1109.21,-157.55)	(322.47,1361.90)	(320.00,180.00)	627.02	-27.95	-0.11	35.37	44.56	27.53	44581	9153	25018
11	(-10.62,-81.41)	(821.99,90.05)	(290.06,1647.07)	(602.59,127.91)	683.75	-25.63	11.62	40.53	51.42	32.19	160240	35226	35006
11_2	(-10.62,-81.41)	(1629.40,-47.45)	(290.06,1647.07)	(320.00,180.00)	626.26	-23.14	1.17	41.19	54.48	30.23	29817	3911	13385
							Real:	**39**	**50**	**29**			

Table 3.2: Auto-calibration data for a selected scene 1.



(a) Case 1

(b) Case 1_2

(c) Case 2

(d) Case 3

(e) Case 4

(f) Case 6

(g) Case 7

(h) Case 8

(i) Case 9

(j) Case 10

(k) Case 11

(l) Case 11_2

Figure 3.30: Graphic results of Test 1.

Test 2

The second scenario is presented in Figure 3.31, where three points are also selected to measure their distance from the camera. Their values from Google Maps are $A = 24m$, $B = 33m$ and $C = 29m$. The numbers correspond to the vehicle indexes for the volume of the projected prisms comparison.

To analyse the results obtained in the test, Table 3.3 summarizes all the values extracted and computed by the system. In this scenario, there is no perpendicular motion component, so the cases related to a perpendicular intersection (3, 8 and 10) are discarded. Figure 3.32 shows the graphic result of the vehicle prism projection for the manual vanishing point extraction case (groundtruth), and Figure 3.33 depicts the graphic results for the rest of available cases.

The partial conclusions after analysing the graphical and numerical results are explained in the following paragraphs:

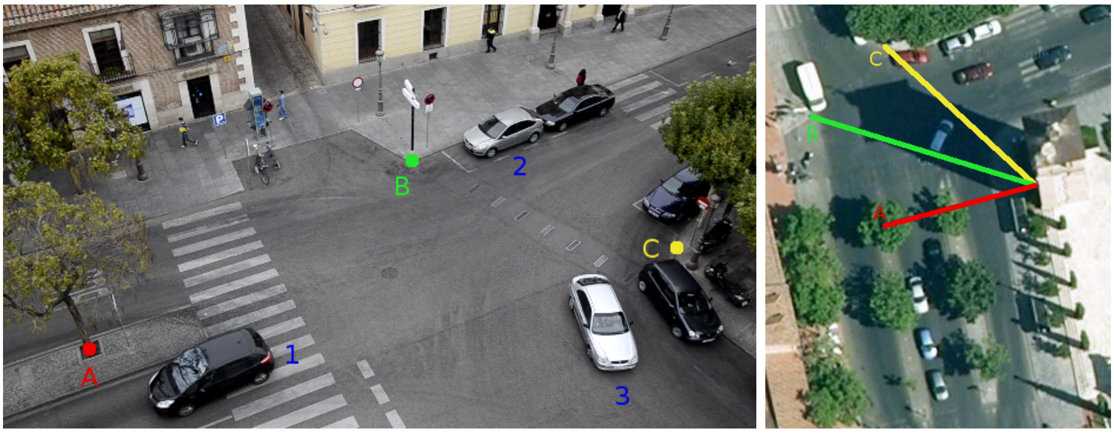


Figure 3.31: Scenario of test 2 and selected points to measure their distance from the camera (located in the tower).

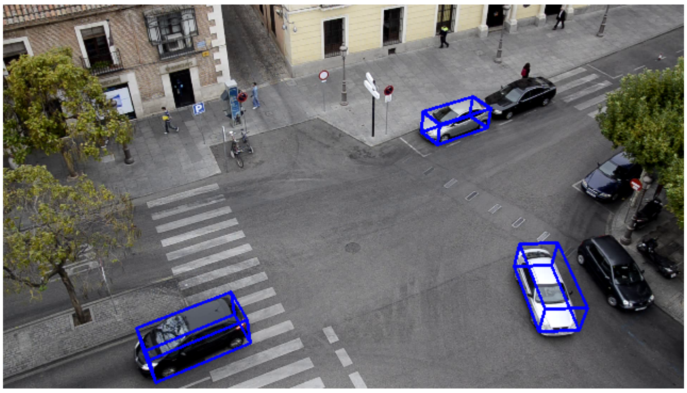


Figure 3.32: Graphic result for the manual vanishing point extraction case.

- In a similar way to the Test 1, the new case 1_2 improves the results obtained by the original version for the same reasons explained before.

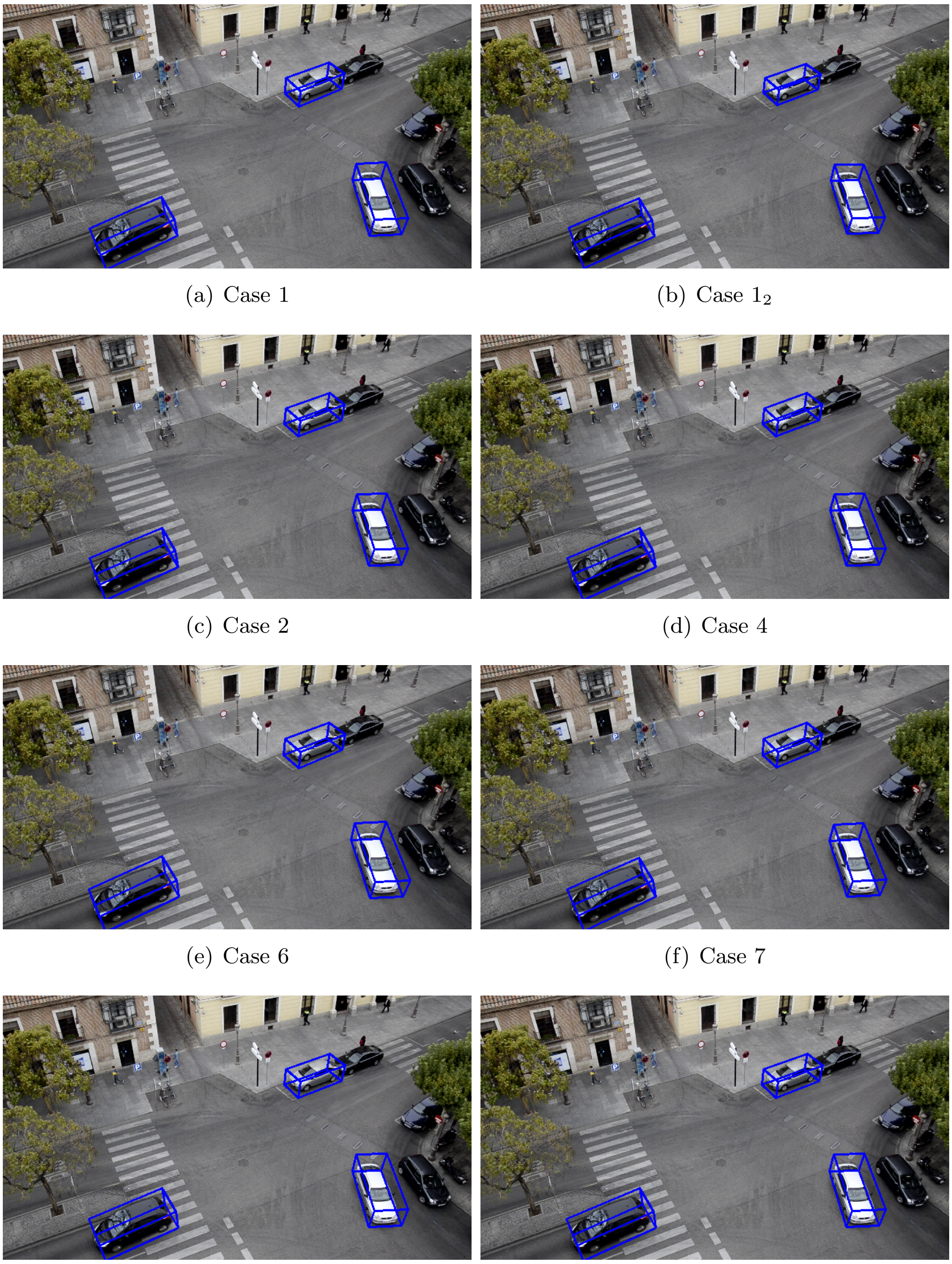
- There are not three strong orthogonal sets of parallel lines, so case 11 is not available. However, it is possible to extract two sets of lines and assume the principal point as the center of the image to calibrate the system (case 11_2). In this case the result is acceptable.

- Related to the measured distances, the computed results are very close to the values provided by Google Maps, which validates the calibration process. Curiously, the groundtruth calibration fits better the 3D prisms but has a higher distance error to the selected points, due to its smaller pitch angle computed.

- In terms of 3D prisms projections, and as can be observed graphically, all the cases are pretty similar and acceptable. The height of the prisms is a little bit higher than desired, but due to the high pitch values it does not affect in case of severe occlusions.

- Once again, the methods that extract the principal point through zooming process are more accurate. It is a small numerical and graphical difference, but the zooming option provides robustness against unexpected situations of the scene in case one vanishing point is unavailable.

- In the two experiments, three points and three volumes have been analysed, covering as many possible cases. The location is similar in both situations, avoiding the center of the image because it is less affected by the extrinsic parameters. Instead of that the challenging positions were taken. The good results demonstrate the effectiveness of the algorithm.



(a) Case 1

(b) Case 1_2

(c) Case 2

(d) Case 4

(e) Case 6

(f) Case 7

(g) Case 9

(h) Case 11_2

Figure 3.33: Graphic results of Test 2.

CASE	VP1	VP2	VP3	OC	FOCAL	PITCH	ROLL	distA	distB	distC	vol1	vol2	vol3
12/5	(1743.01,-253.81)	(-44.74,-257.78)	(320.48,931.17)	**(321.67,182.49)**	**568.84**	**37.23**	**0.13**	**28.93**	**38.88**	**34.40**	**65291**	**18220**	**56644**
1	(2070.26,-353.94)	(-15.06,-321.74)	(331.56,807.11)	(322.25,184.02)	592.58	43.57	-0.88	26.38	33.05	30.04	87171	34655	84796
1_2	(2070.26,-353.94)	(-46.96,-359.50)	(320.31,835.50)	(322.25,184.02)	588.29	42.10	0.17	26.83	34.17	30.99	86045	31662	75565
2	(1146.65,-380.17)	(-515.12,-385.10)	(320.31,835.50)	(322.25,184.02)	604.57	42.88	0.18	26.60	33.51	30.52	92121	34055	85284
3	–	–	–	–	–	–	–	–	–	–	–	–	–
4	(2113.44,-377.46)	(-60.93,-380.32)	(321.05,839.24)	(322.25,184.02)	604.70	42.71	0.09	26.67	33.63	30.60	95963	33071	85652
6	(2070.26,-353.94)	(-15.06,-321.74)	(320.31,835.50)	(309.60,157.15)	603.28	41.68	-0.87	28.19	36.41	32.76	97215	39151	92379
7	(2070.26,-353.94)	(-15.06,-321.74)	(329.67,810.68)	(320.00,180.00)	594.08	43.30	-0.88	26.63	33.49	30.40	94343	33758	83715
8	–	–	–	–	–	–	–	–	–	–	–	–	–
9	(1146.65,-380.17)	(-503.72,-380.00)	(320.31,835.50)	(320.00,180.00)	605.64	42.75	0.00	26.81	33.86	30.80	92121	36799	85715
10	–	–	–	–	–	–	–	–	–	–	–	–	–
11	–	–	–	–	–	–	–	–	–	–	–	–	–
11_2	(2113.44,-377.46)	(-55.73,-373.71)	(321.05,839.24)	(320.00,180.00)	607.33	42.66	-0.08	26.86	33.92	30.84	92121	36801	85714
							Real:	**24**	**33**	**29**			

Table 3.3: Auto-calibration data for a selected scene 2.

3.6 Analysis of vanishing point sensitivity

In this section, a sensitivity study of the vanishing points is presented. For this purpose, an example of auto-calibration is done (see Figure 3.34) and the coordinates of each vanishing point are modified in ±200 pixels to appreciate the impact over the computed parameters. The pixel variation is represented by the violet squares around each vanishing point. As the smaller vanishing point is located in $\mathrm{VP}_1 = (-123, -132)$, ±200 pixels means a variation of 162% and 151% respectively.

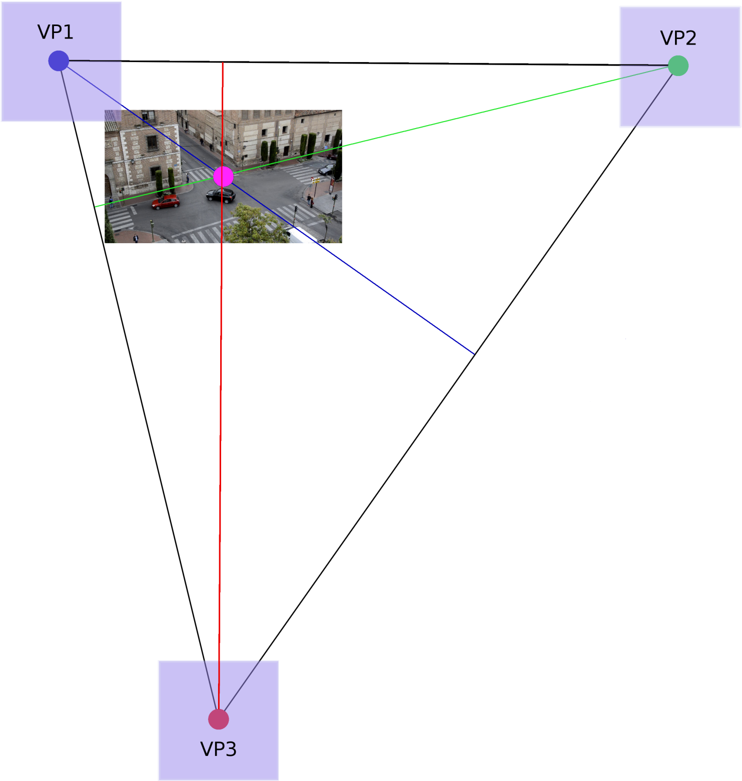


Figure 3.34: Calibration example to analyse the VP sensitivity.

The graphics represented on Figures 3.35 have been obtained varying the horizontal coordinate of VP_1 in ±200 pixels. After that, Figure 3.36 represents the results of varying the vertical coordinate the same amount of pixels. All charts have a symmetrical component around VP±0, so the effect of the positive and negative variation of the vanishing point is almost similar.

The conclusions for each graph are described in the next paragraphs:

- OC. error: is the Euclidean distance (in pixels) between the principal points with and without coordinate variation. The effect is similar in horizontal and vertical changes, but stronger for the vertical ones.

- Focal error: is the percentage error in focal distance against the original value. The effect is significantly stronger for the vertical changes.

- Pitch error: is the angle difference between the pitch before and after varying the coordinates of the point. The effect is almost the same in both axes.

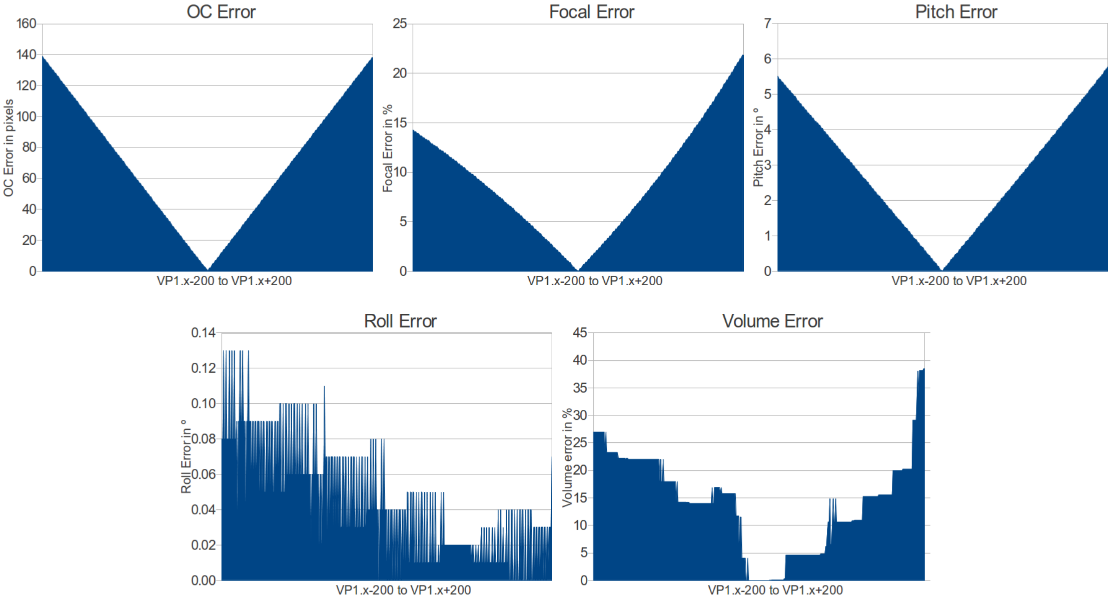


Figure 3.35: Horizontal sensitivity analysis in VP$_1$.

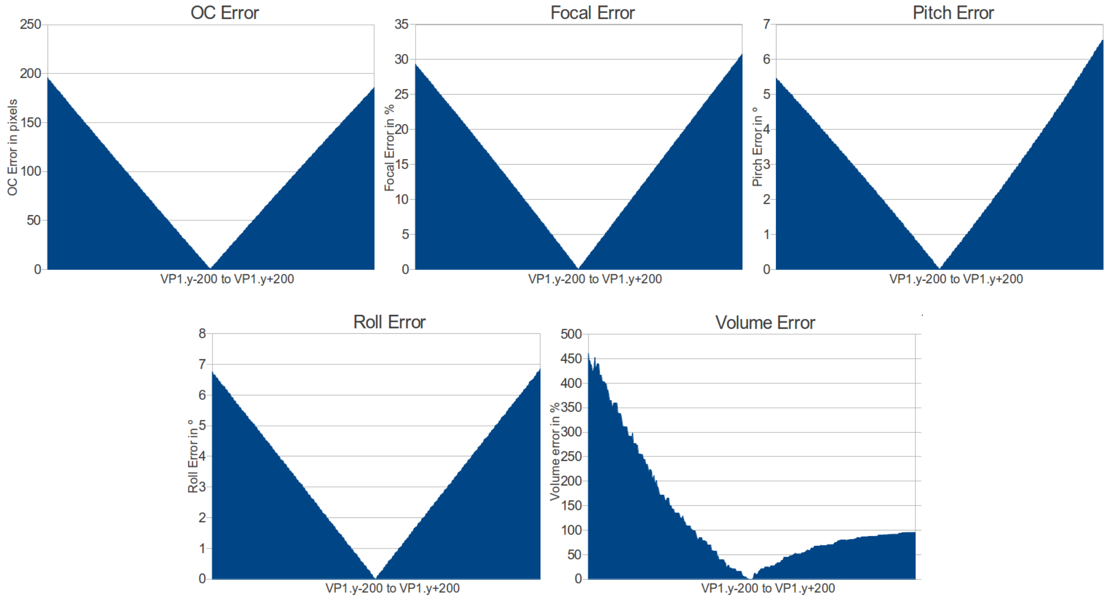


Figure 3.36: Vertical sensitivity analysis in VP$_1$.

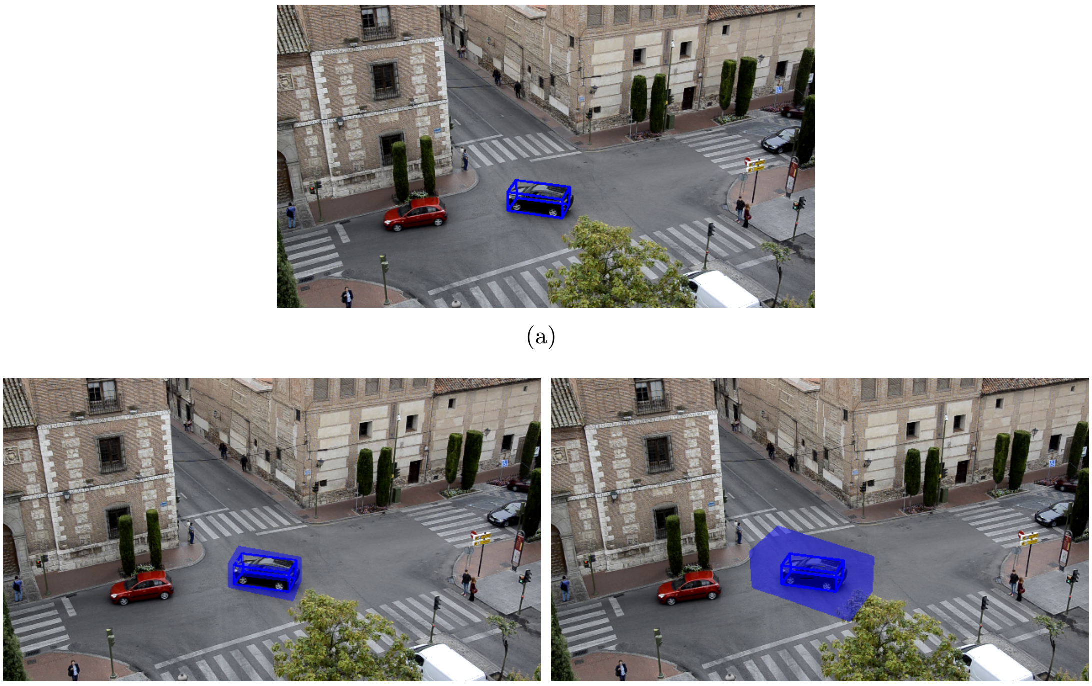
- Roll error: is the angle difference in degrees between the roll obtained after varying the coordinates of the point and the original value. The equation to compute the roll angle is defined by:

$$roll = \tan^{-1}\left(\frac{u_3 - u_0}{v_3 - v_0}\right) \tag{3.24}$$

Therefore the effect of varying an horizontal vanishing point is caused only through the principal point variations. In the case of an horizontal variation, the vanishing point has a vertical influence over the principal point and u_0 remains practically unaltered. Hence, the numerator of the equation does not vary, and as the vertical component of VP_3 is very high (1646 pixels) the variations of v_0, and consequently the roll, are not too significant.

In contrast, in the case of a vertical variation, the vanishing point has a horizontal influence over the principal point and v_0 remains practically unaltered. Hence, the numerator of the equation has a considerable change, reflected in the computed angle. These effects are clearly represented in the corresponding graphics.

- Volume error: is the percentage error committed in the projected volume of a vehicle. Figure 3.37(a) depicts an example of volume projection over a vehicle with the groundtruth calibration parameters. Figures 3.37(b) and 3.37(c) show a combination of all projected volumes for the variations studied. The images demonstrate graphically the impact of the coordinate variations with a violet area.



(a)

(b)

(c)

Figure 3.37: Volume vehicle projection comparative in an horizontal vanishing point variation. (a) Groundtruth volume projection by manual calibration. (b) Volume projections due to horizontal coordinate variations. (c) Volume projections due to vertical coordinate variations.

The case of VP_2 is almost the same than VP_1 but horizontally less sensitive, because its horizontal coordinate is very large (1550 pixels) and the impact of a 200 pixels variation is smaller. And the case of VP_3 is pretty similar to VP_2, but exchanging the influences as a vertical vanishing point, i.e. a small influence in vertical variations and a big influence in horizontal variations.

The selected vanishing point VP_1 is the most sensitive to a change of its coordinates, so it will stablish the variation limits. If a 25% of volume error is considered the maximum acceptable variation, the ranges are $\mathrm{VP}_x \pm 180$ and $\mathrm{VP}_y \pm 30$ pixels. This maximum range is represented on Figure 3.38 by the transparency around the car.

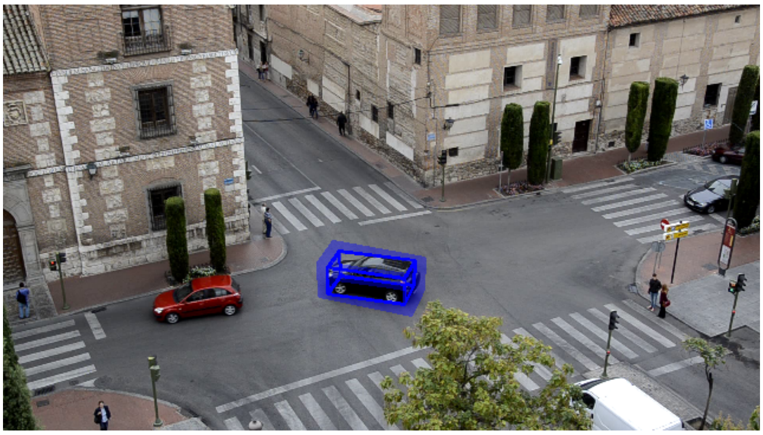


Figure 3.38: Projected vehicle volume under maximum VP range variations.

3.7 Conclusions

In this chapter, an auto-calibration camera approach has been proposed through an hierarchical algorithm based on the scene. It is an important step for the final goal of the thesis, because it provides very useful information to compute the object sizes, necessary for the algorithm proposed in the next chapter.

The performance of the method has been described through the results obtained by two selected videos. 30 more sequences from different scenarios and conditions have been used to test the developed auto-calibration methods. As a result, a comparative table (Table 3.4) has been constructed with the average errors of the main intrinsic and extrinsic parameters extracted (focal distance, pitch and roll).

CASE	FOCAL (%)	PITCH (°)	ROLL (°)
12/5	0.00	0.00	0.00
1	3.85	2.08	0.52
1_2	**2.29**	**1.68**	**0.30**
2	4.69	2.83	0.34
3	8.14	3.55	0.65
4	6.68	3.05	0.67
6	3.52	1.46	0.51
7	3.88	2.05	0.51
8	4.05	2.25	0.69
9	4.40	2.57	0.26
10	7.47	3.11	0.64
11	9.20	2.46	2.11
11_2	7.18	3.16	0.65

Table 3.4: Auto-calibration errors comparative table.

As can be seen, case 1, and its improvement above all (case 1_2), are the best solutions due to the strong parallel component of their orthogonal elements and the zooming chance. Near them, cases 6 and 7 have similar results. It was expected because they are the relative cases to the first one, but without zoom. On the other hand, the worst options (although graphically acceptable in most situations for the system proposed), are cases 3, 4, 10 and 11, based on perpendicular intersections (not always available or strictly perpendicular), and structured scenes (not always with strictly orthogonal components).

The obtained results are really satisfactory: the low error of the 3D prisms projections and distance measurements proves the strength of the system, and the multiple options of the hierarchical tree provide high versatility to cover most of the possible traffic scenarios. Furthermore, the system is able to adapt the calibration parameters in case of PTZ camera displacements without manual supervision.

Even if there is no chance to auto-calibrate the camera (due to absence of orthogonal components), the manual input of lines remains as a valid option which allows the user to control the system in a short time.

Finally, the acceptable variation ranges of the vanishing points coordinates (studied in the sensitivity analysis), give the algorithm a tolerance of at least 30 pixels, which means that small errors are not critical.

Chapter 4

Target detection and tracking

4.1 Introduction

After calibrating the camera, an approximate size of pedestrians and vehicles in the image can be obtained using a standard size for them in world coordinates. This step will give the system a notion of how big are the searched elements. In this chapter, a multilevel framework to detect and track pedestrians and vehicles is presented. Figure 4.1 illustrates the flowchart of the proposed framework, which consists of 4 levels: 1) *Image segmentation level*, to create and handle a background model and to obtain the foreground objects; 2) *features level*, which extracts and follows features of the foreground objects; 3) *clustering level*, which is in charge of managing occlusions and create object clusters; and 4) *tracking level*, which tracks all the segmented objects.

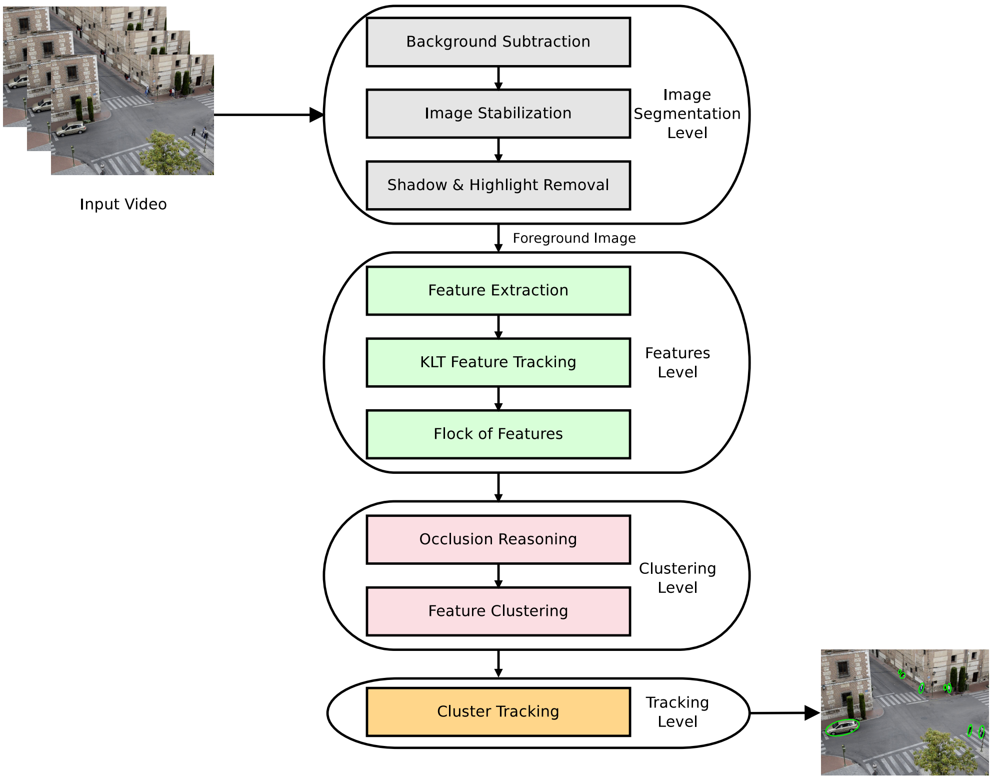


Figure 4.1: Flowchart of the proposed framework.

4.2 Image segmentation

Traffic surveillance systems consist on detecting and tracking targets by a static camera. In this context, *background subtraction* reveals as the best solution to segment the moving objects of the image. However, although pedestrians and vehicles are generally the only objects which are moving in the field of view, the algorithm is susceptible to instabilities of the camera and both global and local illumination changes, so a detection of these problems is needed to achieve satisfying results. Therefore, the complete object segmentation algorithm consists of the following steps: *background subtraction*, *image stabilization* and *cast shadows and illumination changes detection.*

4.2.1 Background subtraction

The basic idea of background subtraction is to subtract the current image from a reference image that models the background scene. Rather than explicitly modelling the values of the pixels as one particular kind of distribution, each pixel is modelled by a mixture of K Gaussian distributions (Gaussian Mixture Model or GMM), whose mean and variance is adapted over time.

The probability that a certain pixel has a value X_t at time t can be written as:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{4.1}$$

where the mean $\mu_{i,t}$, covariance $\Sigma_{i,t}$ and weight $\omega_{i,t}$, are the parameters of the i^{th} gaussian component, and η is the gaussian probability density function described by:

$$\eta(X, \mu, \sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1} (X-\mu)} \tag{4.2}$$

For computational reasons the covariance matrices are isotropic. This assumes that the red, green, and blue pixel values are independent and have the same variances $\sigma_{i,t}^2$. While this is certainly not the case, the assumption allows to avoid a costly matrix inversion at the expense of some accuracy:

$$\Sigma_{i,t} = \sigma_{i,t}^2 I \tag{4.3}$$

Given a new data sample X_t, the recursive equations to update the model are: [17]

$$\omega_i = \omega_i + \alpha(\theta_i - \omega_i) \tag{4.4}$$

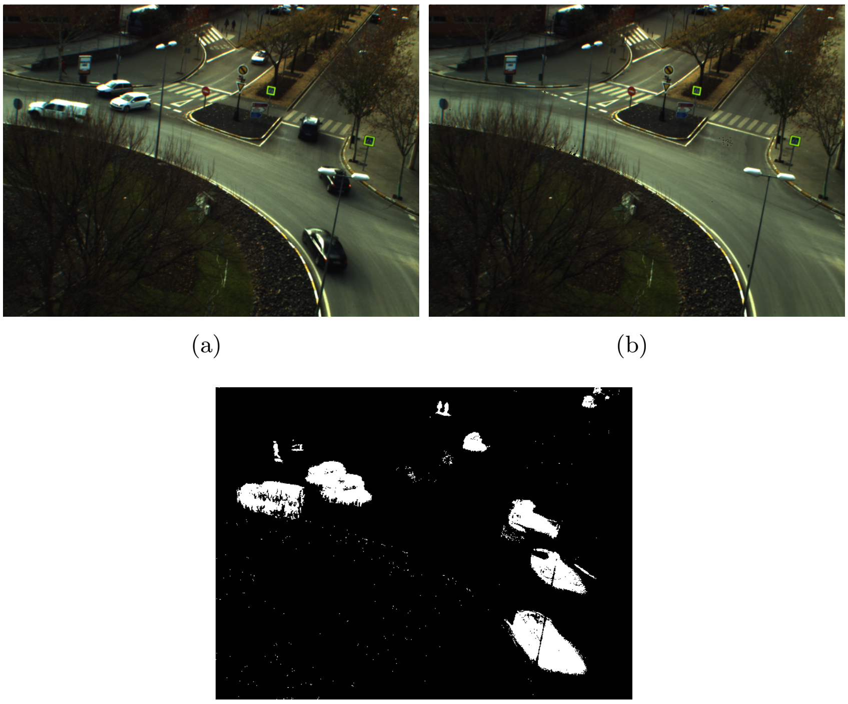
$$\mu_i = \mu_i + \theta_i(\frac{\alpha}{\mu_i})\delta_i \tag{4.5}$$

$$\sigma_i^2 = \sigma_i^2 + \theta_i(\frac{\alpha}{\mu_i})(\delta_i^T \delta_i - \sigma_i^2) \tag{4.6}$$

where α is the learning rate and $\delta_i = X_t - \mu_i$. For a new sample the ownership θ_i is set to 1 if the sample matches with a component of the mixture (sorted by the value of $\frac{\omega}{\sigma}$)

and 0 for the remaining models. The matching is defined by the Mahalanobis distance between the sample and the gaussian component of the mixture and a threshold. If there is no matching, a new component is generated with $\omega_{i+1} = \alpha$, $\mu_{i+1} = X_t$ and $\sigma_{i+1} = \sigma_0$, where σ_0 is a predefined initial variance. If a predefined maximum number of components has been reached, the component with the smallest weight is discarded.

Figure 4.2 shows the result of this step: the original image, the modelled background, and the extracted foreground. Usually, the intruding foreground objects are represented by gaussians with small weights. Therefore, it is possible to approximate the background model by the component with largest weight. In the case of the foreground extraction, although there are no strong shadows, they are labelled as foreground due to the light change they produce in the asphalt.



(a) (b)

(c)

Figure 4.2: Background subtraction result on a tested sequence. (a) Original image. (b) Modelled background. (c) Extracted foreground.

To manage new elements in the image and background model, the adaptation process is straightforward. For example, if a new object comes into a scene and remains static for some time, it will be temporally presented as an additional gaussian component. Since the old background is occluded, the weight of the new gaussian will be constantly increasing and the old one decreasing. If the object remains static long enough, its weight becomes larger and it can be considered to be part of the background.

One of the significant advantages of this method is that when something is allowed to become part of the background, it does not destroy the existing model. The original background color remains in the mixture until it becomes the k^{th} most probable gaussian component and a new color is observed. Therefore, if an object is stationary just long enough to become part of the background and then it moves, the distribution describing the previous background still exists with the same μ and σ^2, but with a lower weight, and will be quickly reincorporated into the background.

4.2.2 Image stabilization

Most of the traffic monitoring systems entail the use of cameras in outdoor environments. Because of that, they are exposed to vibrations and shaking due to wind and other inclemencies, which can cause visible frame-to-frame jitter and associated foreground errors. To avoid the mentioned problems, an image stabilization module has been developed. It captures the movement of static feature points, extracted and matched with SURF [45], between the current image and the background model, to estimate the camera displacement. After extracting these points, the neighbourhood of each one is represented by a feature vector and matched between the images, based on Euclidean distance. In case of erroneous measurements or incorrect hypotheses about the interpretation of data, RANSAC is used to filter the outliers. After RANSAC, SURF feature pairs are used to compute the homography matrix between both images. Finally a perspective transformation based on this homography matrix is applied to the current image to compensate the movement. The result of the image stabilization step can be seen in Figure 4.3.

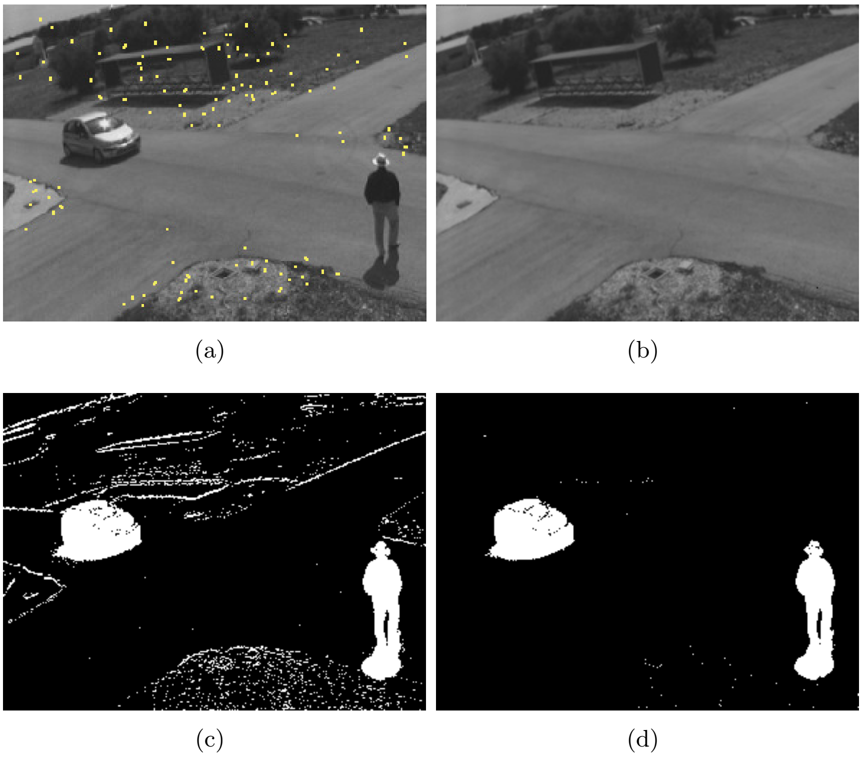


Figure 4.3: Result of image stabilization. (a) Original image with camera shake and SURF points. (b) Modelled background. (c) Extracted foreground without stabilization. (d) Extracted foreground after stabilization.

After checking the good results provided by this technique, the idea was extrapolated to detect camera motion in case of using a PTZ camera. As a result, if the detected motion is bigger than a simple shaking (experimentally established with a threshold), the movement is classified as yaw, pitch, roll or zoom displacement and the background model is restarted. To differentiate between angle and zoom variations it is enough to analyse the direction of the motion vectors. In the case of angle variations the vectors are parallel while for zoom variations they are concurrent.

Figure 4.4 illustrates the effect of a yaw (pan) angle change, with parallel and horizontal motion vectors. In this case, the calibration parameters remain constant and it is not necessary to recalibrate. However background subtraction needs to be restarted.

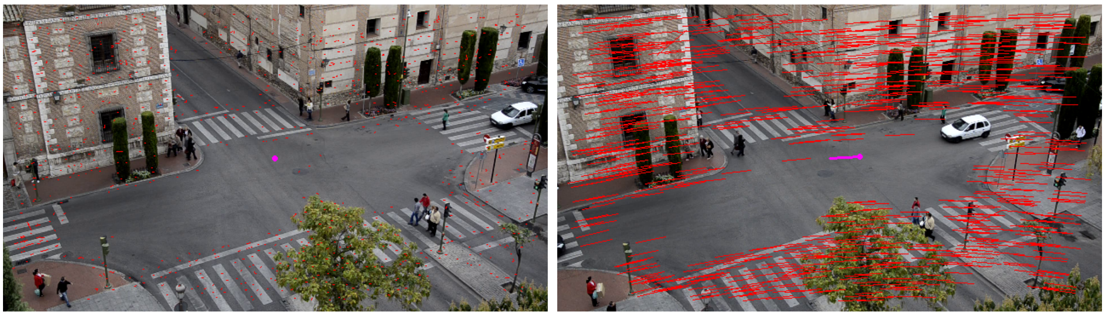


Figure 4.4: Example of detected yaw change. (a) Source image before the yaw variation. (b) Image after yaw variation and motion vectors. The purple line represent the average motion.

Figure 4.5 represents the effect of a pitch (tilt) angle change, with parallel and vertical motion vectors. In this case, background subtraction also needs to be restarted, and the calibration parameters are not constant so it is necessary to recalibrate. However, if the pitch equation is analysed, reminded in Equation (4.7), the only parameter that changes in a pitch variation is the vanishing point V_y. Therefore, to recalibrate the camera, a search of the vertical vanishing point is enough.

$$pitch = tan^{-1}\left(-\frac{f\sin\gamma}{u_{v_y} - u_0}\right) \tag{4.7}$$

where f is the focal length of the camera, γ is the roll and u_{v_y} and u_0 are the horizontal coordinates of the vertical vanishing point and principal point respectively.

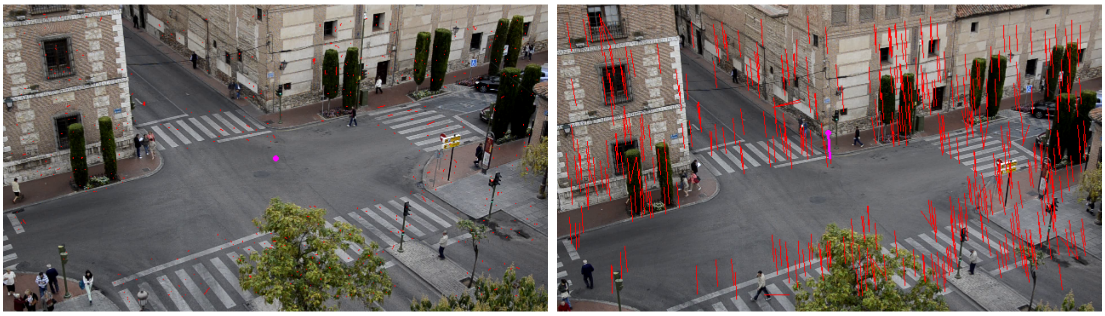


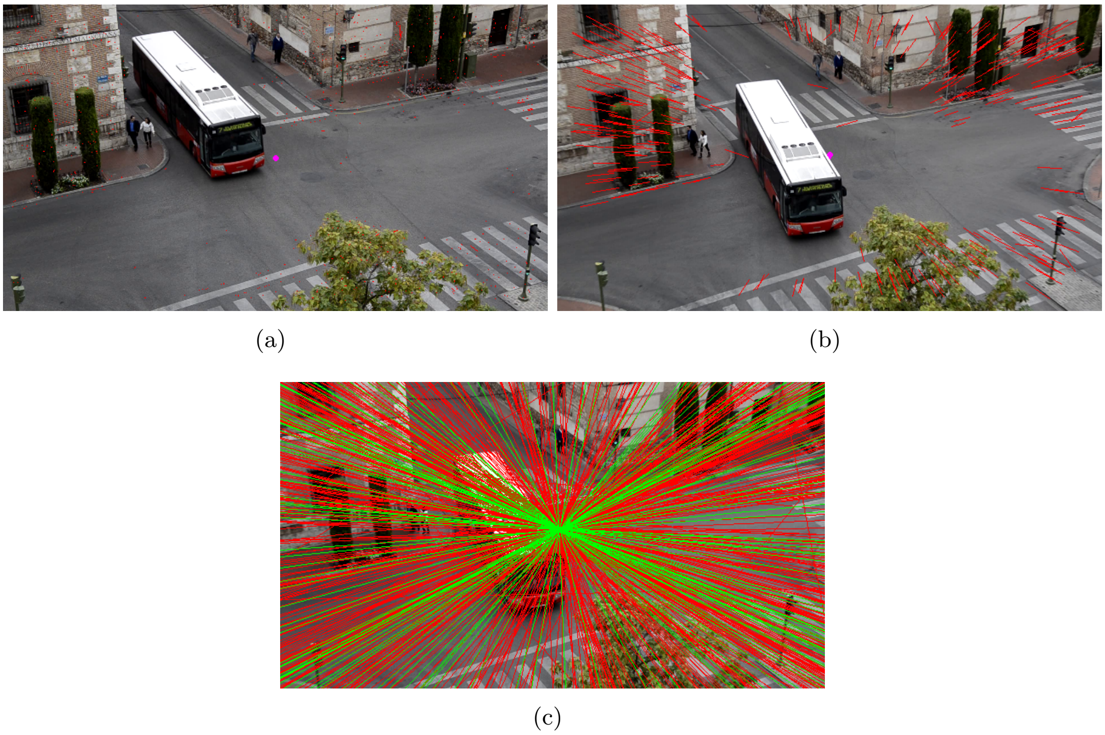
Figure 4.5: Example of detected pitch change. (a) Source image before the pitch variation. (b) Image after pitch variation and motion vectors.

Figure 4.6 represents the effect of a zoom change with its concurrent motion vectors. In this case as well, background subtraction needs to be restarted and, as the calibration parameters are not constant, camera recalibration has to be done. However, the parameter which changes with a zoom variation is the focal length, so not all the variables need to be computed. If its expression, reminded in Equation (4.8), is analysed, only the vanishing points V_y and V_x vary. Therefore, to recalibrate the camera, a search of these two vanishing points is enough.

The principal point is intrinsic of a zoom variation, as explained in Subsection 3.4.3, and with the proposed technique it is possible to extract this parameter. The vectors are projected into lines and the same algorithm developed for the vanishing point estimation (based on RANSAC) is used to compute the intersection point.

$$f = \sqrt{(\sin\gamma(u_{v_x} - u_0) + \cos\gamma(v_{v_x} - v_0))(\sin\gamma(u_0 - u_{v_y}) + \cos\gamma(v_0 - v_{v_y}))} \tag{4.8}$$

where γ is the roll, (u_{v_x}, v_{v_x}) and (u_{v_y}, v_{v_y}) are the two necessary vanishing points and (u_0, v_0) is the principal point of the image.


(a)

(b)

(c)

Figure 4.6: Example of detected zoom change. (a) Image before zooming. (b) Image after zooming and motion vectors. (c) Principal point extraction.

As explained in Subsection 3.3.2, red lines are the outliers and green lines are the inliers, concurrent into the searched principal point.

To summarize the information previously described, Table 4.1 depicts which changes or parameters are necessary in each particular case of a camera displacement. 2 VP's means any two orthogonal vanishing points of the image and OC stands for the principal point.

	Yaw	Pitch	Zoom	Pitch + Zoom
Background	Restart	Restart	Restart	Restart
Calibration	None	V_y	2 VP's + OC	2 VP's + OC

Table 4.1: Summary of necessary changes after camera displacement.

4.2.3 Cast shadows and illumination changes detection

Background subtraction step detects all the moving objects that do not belong to any component of the mixture. Despite the robust detection in good illumination conditions, the algorithm suffers with the presence of shadows and sudden illumination changes. For this reason, a shadow and highlight detection algorithm is needed.

As described in the state of the art chapter, there is no single robust shadow detection technique and it seems better for each particular application to develop its own algorithm according to the nature of the scene. The objective of this thesis is not finding the final shadow detection method, therefore, the developed algorithm has been chosen to work correctly with the author's dataset and the generalization for all possible situations and conditions has been discarded. Considering this idea, the principle used for the technique is based on the fact that a shadow or a highlight changes color properties of the objects, but not their surface properties such as texture. The method is characterized by a region-level analysis in spite of a pixel-level, hence decreasing the sensitiveness to image noise. The technique used is the normalized cross correlation, and particularly *Color Normalized Cross Correlation* (CNCC). The algorithm uses this method to compare the texture of every foreground pixel, by a neighbourhood window, with the correspondent one in the background model.

Let B be the background image and I an image of the video sequence. Then, considering for each foreground pixel a $(2N + 1)$ window, the NCC between the image and the background is given by Equation (4.9). In the case of a color image, template summation in the numerator and each sum in the denominator is done over all of the channels, with separate mean values used for each channel.

$$NCC = \frac{E_t}{E_B E_I} \tag{4.9}$$

$$E_t = \sum_{n=-N}^{N} \sum_{m=-N}^{N} B(n,m) I(n,m) \tag{4.10}$$

$$E_B = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} B(n,m)^2} \tag{4.11}$$

$$E_I = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} I(n,m)^2} \tag{4.12}$$

For a pixel with an illumination change but similar texture, correlation is very close to 1. Otherwise it is close to 0, so the threshold parameter is not critical. Moreover, in the case of shadows, the energy E_I has to be lower than E_B. After removing pixels with the NCC thresholding, a closing operation is done to fill small holes in the contour.

Two different space colors are chosen together to compute the correlation. On the one hand, RGB is used for soft shadows and sudden illumination changes; and on the other hand, for strong shadows the international standard CIE 1931 XYZ color space has been tested empirically with better results; so two different matching analysis are done. Figures 4.7 and 4.8 show the result of removing soft shadows in dusk conditions and strong shadows in a sunny day. Finally Figure 4.9 depicts the result of removing a sudden illumination variation.

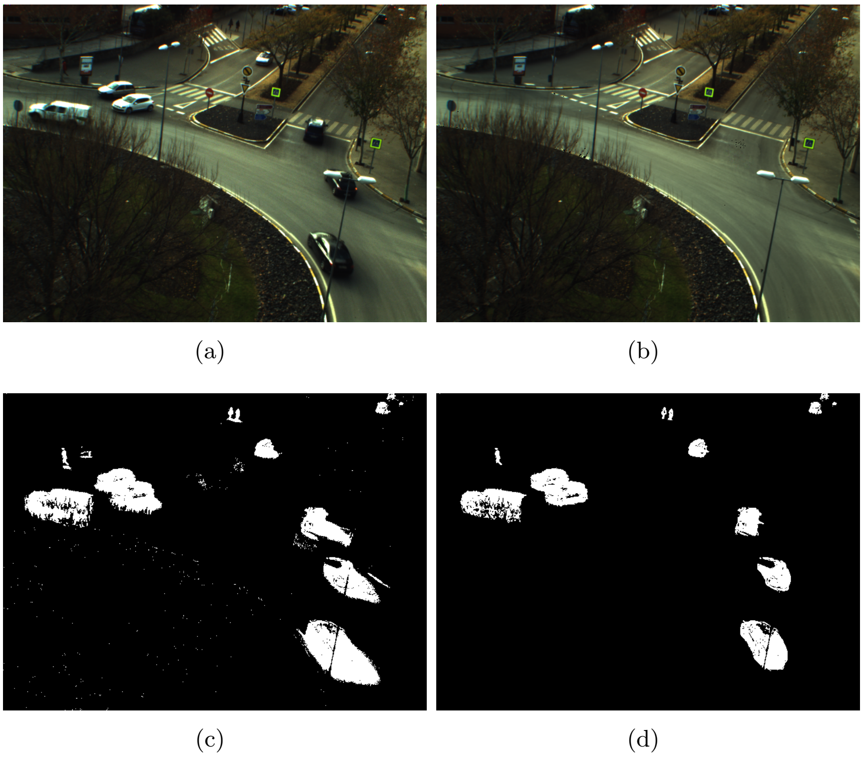


Figure 4.7: Soft shadow removal. (a) Original image with shadows. (b) Background model. (c) Initial foreground. (d) Foreground after shadow removal.

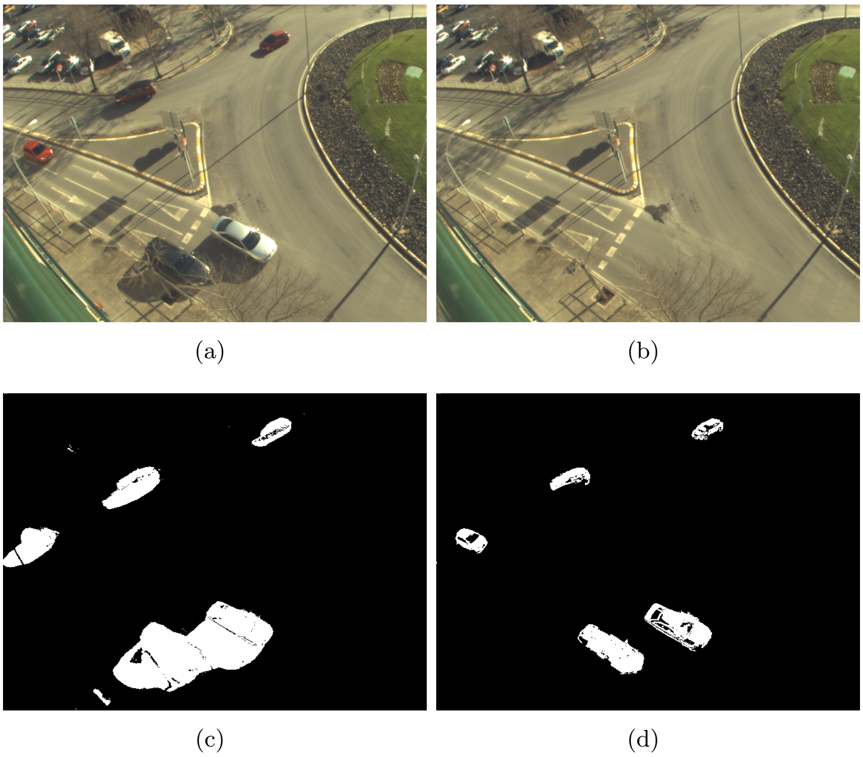
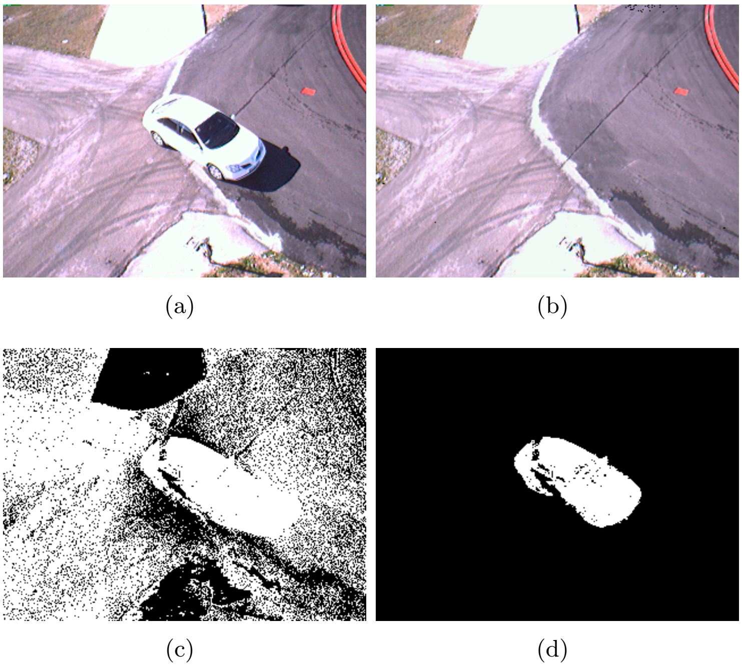


Figure 4.8: Hard shadow removal. (a) Original image with shadows. (b) Background model. (c) Initial foreground. (d) Foreground after shadow removal.



(a) (b)

(c) (d)

Figure 4.9: Global illumination change managed. (a) Original image with highlight. (b) Background model. (c) Initial foreground. (d) Final foreground.

Avoiding the problems of this technique due to the absence of imaging scale, rotation, and perspective distortions, the method works fairly good in every tested situations, under different illumination conditions. However, sometimes the algorithm falsely labels pixels with lower luminance value than the background but similar chromaticity. To solve this problem, a similar solution that the one applied in the background subtraction stage is adopted. Rather than explicitly classifying the pixels with one particular kind of distribution, a multidistribution statistical learning process is used [28].

The appearance of a shadowed surface shows a certain regularity even in scenes with complex illumination conditions. This regularity is caused by several factors: the light sources are generally stable and fixed, the foreground objects moving in the scene have a similar scale factor, and they move following physical constraints like walls, ground, roads, hallway, etc. Since different foreground objects block light sources in a similar way, the shadows cast on the background surfaces are relatively similar at the pixel level. This phenomenon is particularly strong in busy hallways or highways where different people or different vehicles induce the same intensity variation on a surface when blocking a light source. The repetitiveness of the appearance of cast shadows is exploited to learn shadowed surface values. This is done by parametrizing probability density functions representing these shadowed surfaces.

First, a weak classifier based on the normalized cross correlation explained before is used. After that, all pixels considered as shadow are added to a multidistribution learning algorithm, once again the Gaussian Mixture Model. In this implementation it is called Gaussian Mixture Shadow Model (GMSM). The GMSM is composed of learned distributions representing background surfaces when shadows are cast on them. This continuous learning process is quite similar than the background subtraction one. For each frame of the image sequence, a pixel is labelled as a moving cast shadow if its value can be associated with one of the distributions stored in the GMSM at that time.

4.3 Feature extraction and tracking

After extracting the image foreground and removing the shadow and highlight effects and camera displacements, a new step to distinguish between different objects is done. Due to partial and global occlusions, the detected objects could be fragmented, joined with a close one or even lost. Therefore, the foreground blobs are not valid without a high level object extraction and a tracking stage.

Feature-based tracking gives up the idea of tracking objects as a whole, after obtaining the different regions through background subtraction. Even in the presence of partial occlusion, some of the features of the moving objects remain visible, so it gives a chance to overcome the occlusion problem. Furthermore, the same algorithm can be used for tracking in daylight, twilight or night-time conditions, as well as different traffic conditions, camera positions and shape changes, being able to consistently track objects over long sequences. It is self-regulating because it selects the most salient features under the given conditions. The idea of the algorithm is to extract and track foreground features and cluster them into objects using proximity, motion history, speed, orientation and the size constraints provided by the calibration.

The proposed method is called *flock of features* and it is based on the work of Kölsch et al. [46]. The concept comes from natural observation of flocks of birds or fishes. It consists of a group of members, similar in appearance or behaviour to each other, which move congruously with a simple constraint: members keep a minimum safe distance to the others; but not too separated from the flock. This concept helps to enforce spatial coherence of features across an object, while having enough flexibility to adapt quickly to large shape changes and occlusions.

FAST feature extractor [47] combined with pyramid-based KLT feature tracking is chosen as the main tracker where the flock constraints are applied. This combination has been chosen experimentally for its performance and better results against other methods. Features are extracted from the foreground regions and tracked individually frame to frame. Moreover each feature is analysed over time increasing or decreasing a level of life in case of finding or not a matching in the following frames. If this level reaches 0, the feature is removed or reallocated inside the object depending on the constraints of the flock. When a feature has a match in the background image, it is considered invalid and removed. Figure 4.10 depicts an example of a traffic scenario and the FAST features extracted from the foreground image. Figure 4.11 shows a feature tracking example of the same scene over time.

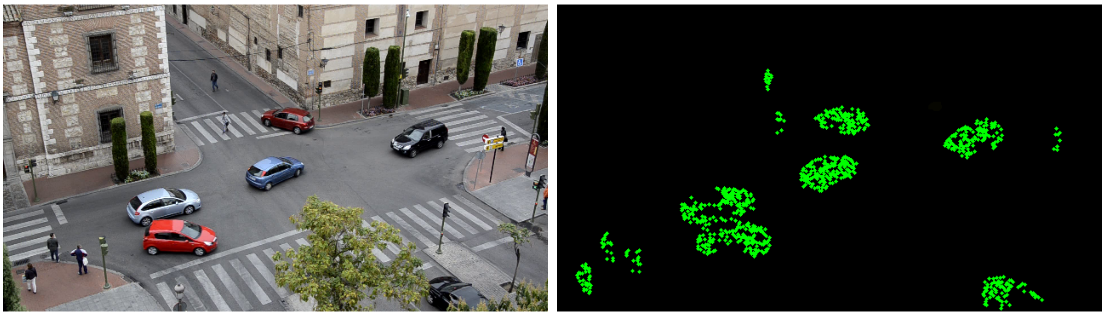


Figure 4.10: FAST features extracted from an image. (a) Source image. (b) Extracted features.

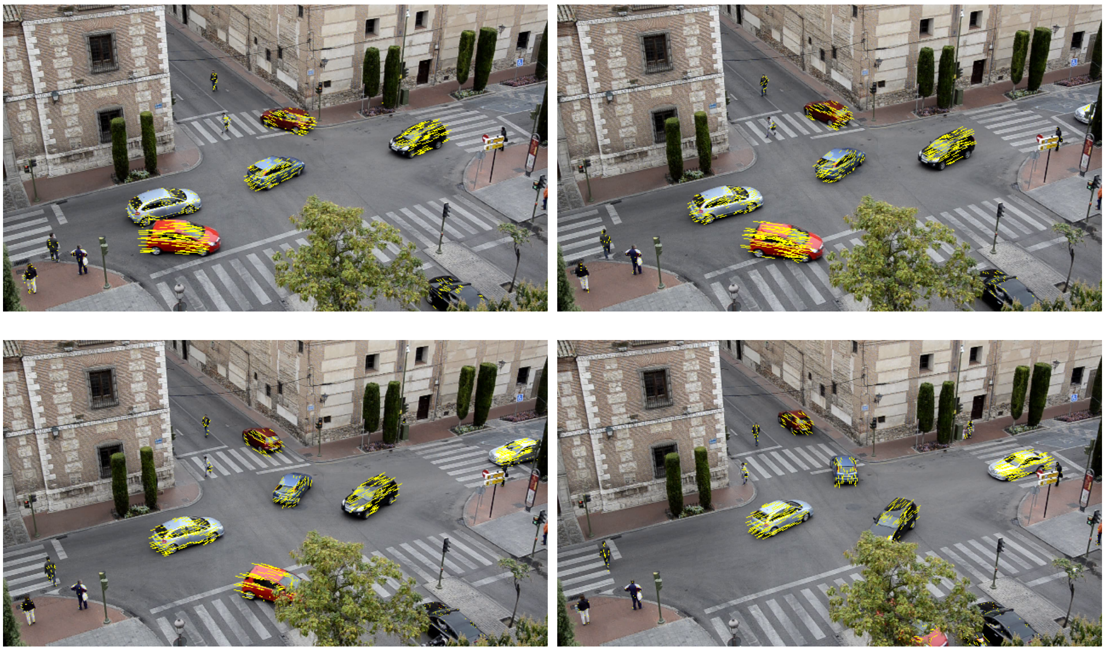


Figure 4.11: Feature tracking sequence.

Finally, the motion of the features (speed and angle) is measured for posterior analysis. The previous feature position and the one estimated by optical flow construct a motion vector. To avoid problems with similar directions but different angles, like 359° and 1°, these vectors are considered as color in the HSV space (Hue=direction, Saturation=speed, Value=1), and then converted into RGB space. Therefore the motion is described by three components. The color associated to the motion vectors of the previous example and the corresponding RGB wheel are shown in Figure 4.12. Table 4.2 depicts four examples of motion conversion to RGB space to clarify the effects.

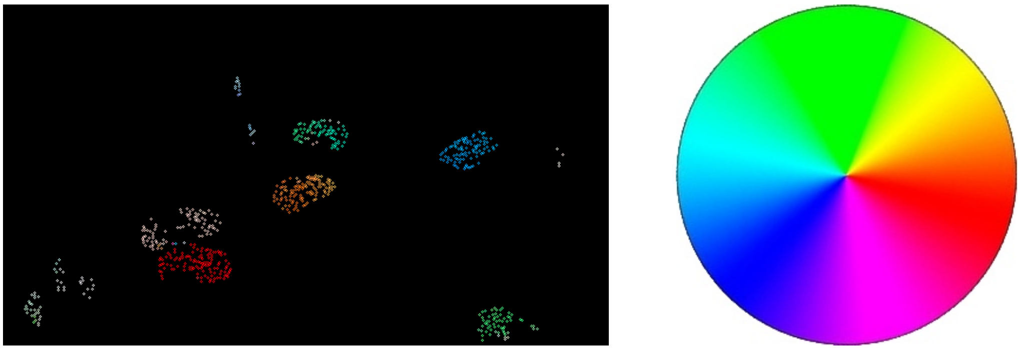


Figure 4.12: Motion vectors considered as color features.

Speed (pix)	Angle (°)	RGB code	RGB color
6	1	[255, 4, 0]	🟥
6	359	[255, 0, 4]	🟥
6	153	[0, 255, 140]	🟩
2	153	[171, 255, 217]	🟩

Table 4.2: Examples of motion conversion to RGB space.

4.4 Clustering

Usually, feature grouping works associating features directly into objects using proximity and motion history. However, the distance between two features that belong to the same object can be much larger than two features that belong to two nearby objects, which can confuse the system. To efficiently deal with the problem, a multilevel grouping algorithm is presented. First, an occlusion reasoning step is done in order to split foreground blobs from different objects. After that, the individual features are associated to a blob and grouped into "small" clusters depending on their motion. Finally these clusters are grouped into object-level ones depending on the 3D sizes and motion.

4.4.1 Partial occlusion reasoning

In computer vision, an occlusion refers to the visual obstruction that an object causes into another due to the perspective view of the camera. It is partial, if some parts of the object remain visible, or global, if the object is not visible. In this section only partial occlusions are studied. Further tracking steps will focus on resolving the global ones.

The first step when considering this problem is to observe the shapes of the objects involved in an occlusion. Figure 4.13 depicts some examples of partial occlusions.

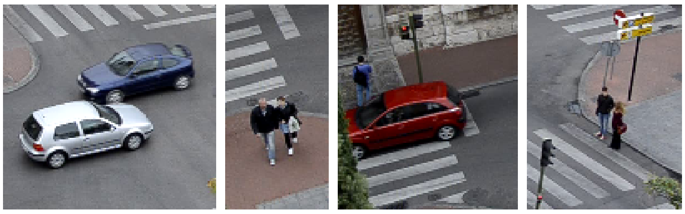


Figure 4.13: Object occlusion examples.

A common characteristic extracted from these images is that the shapes generated by an occlusion are not uniform: non-occluded objects are generally convex, whereas the shape of partially occluded objects become concave. An example of non-occluded and occluded objects is given by comparing their convex hull in Figure 4.14.

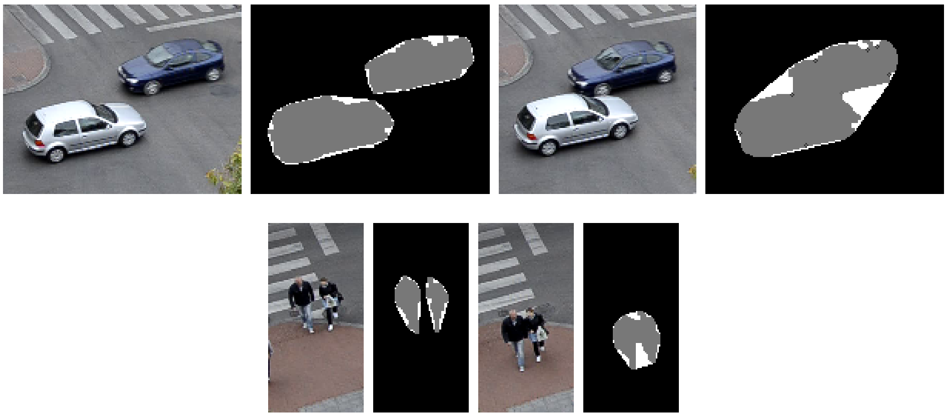


Figure 4.14: Object convex hull examples before and after occlusion. The convex hull is represented in white and the foreground blob in gray color.

It can be seen that non-occluded objects can reach a good fit by their convex hull, which does not hold for occluded objects. Accordingly, if there is an approximate idea of the searched objects sizes, an occlusion can be figured out by studying the blob shapes and their convex hulls. The analysis and description of the object shapes are important topics in pattern recognition and computer vision, and in particular one simple shape descriptor has been widely used in these tasks: the *shape compactness*. It is an intrinsic characteristic of the object shapes defined by:

$$C = \frac{P^2}{A} \tag{4.13}$$

where C is the value of shape compactness, A is the shape area and P is the shape perimeter or boundary length. This way to measure shape compactness is taken from the isoperimetric inequality [48]. The next step to evaluate if a blob is the result of an occlusion is to compare the shape compactness of the object (C_o) and the one of its convex hull (C_h). Obviously C_o is always greater than C_h, because the area of an object is smaller than the one of its convex hull, whereas the boundary length of an object is greater. Therefore, for non-occluded objects C_h is close to C_o, and for occluded ones C_h is smaller than C_o. The ratio between both descriptors is used to discriminate both situations. It is called *compactness ratio* and it is defined by:

$$CR = \frac{C_h}{C_o} \tag{4.14}$$

Another parameter used to detect an occlusion is the *convexity*, and it is determined by the ratio between the areas of the blob and its convex hull as:

$$R_A = \frac{A_o}{A_h} \tag{4.15}$$

where A_o and A_h represent the area of the object and the area of the object's convex hull respectively. Since the denominator is always greater than the numerator, R_A is always less than one. For a non-occluded object its shape is convex and R_A is close to 1, whereas for occluded objects R_A is far less than 1.

The third estimator to consider a blob as an occlusion is its size. After calibrating the camera, the relationship between measures in the 3D world and the image are known. Therefore the approximate sizes of the pedestrians and vehicles located in the ground plane are known. In case of occlusions these sizes will be considerably increased. If the three parameters described above indicate an occlusion, the occlusion reasoning method is run as described in the flowchart of Figure 4.15.

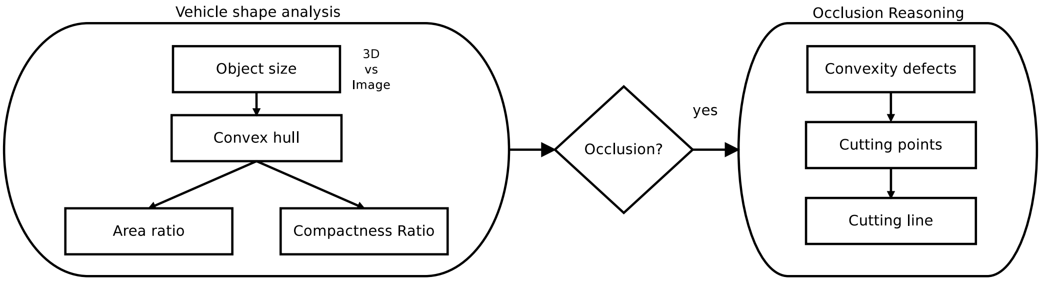


Figure 4.15: Flowchart of the occlusion reasoning method.

An useful way to understand the shape of an object contour is to compute its convex hull and convexity defects. Figure 4.16 illustrates these concepts using the image of a vehicle occlusion. The gray area corresponds to the foreground blob and the coloured areas represent the different defects of the convex hull. Finally, the red marks correspond to the farthest points from the convex hull within each defect, also called *defect points*.

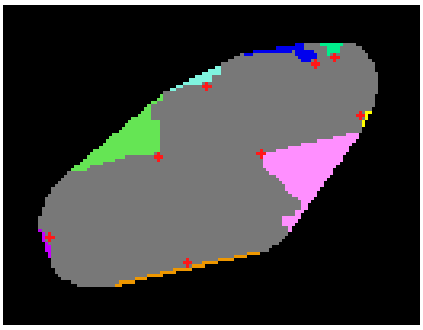
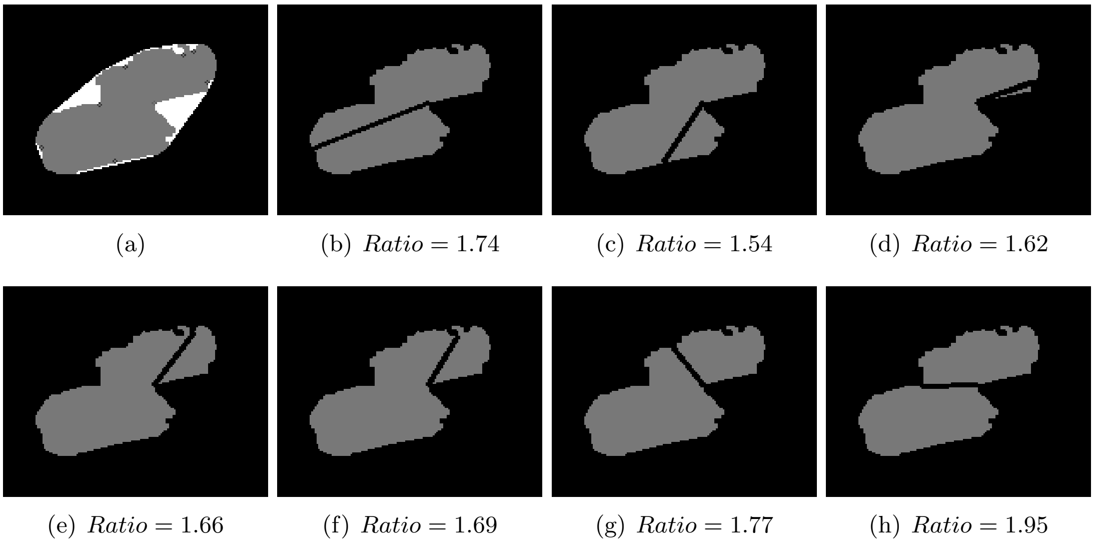


Figure 4.16: Blob, convex hull and convexity defects in an occlusion example.

The distance between the farthest defect point and the convex hull is taken and this point is selected as the first *cutting point*. The next objective is to find an optimum second *cutting point* to create a *cutting line* which separates the blob into two different objects. To extract the second point, the occluded object is sequentially cut by segments that join the cutting point with the rest of defect points. For every line, the area and compactness ratios for each new blob are computed. The chosen *cutting line* is the one that brings the maximum ratio given by the Equation (4.16). Figure 4.17 shows an example of the process to split two occluded vehicles. Each subfigure corresponds to a cutting line and the corresponding ratios are depicted below the images.

$$Ratio = \sum_{i=1}^{2} \frac{R_{Ai} + CR_i}{2} \tag{4.16}$$



(a)

(b) $Ratio = 1.74$

(c) $Ratio = 1.54$

(d) $Ratio = 1.62$

(e) $Ratio = 1.66$

(f) $Ratio = 1.69$

(g) $Ratio = 1.77$

(h) $Ratio = 1.95$

Figure 4.17: Example of computing a cutting line to manage an occlusion.
(a) Initial blob and convex hull. (b)-(h) Different cutting lines and ratios obtained.

Figure 4.18 depicts some examples of occlusion reasoning using the method explained before. As can be seen, vechicle-to-vechicle occlusions, pedestrian-to-pedestrian occlusions and vehicle-to-pedestrian occlusions are correctly managed. This procedure does not require prior knowledge but the known measures from camera calibration. By using this method, most partial occlusions can be effectively handled.

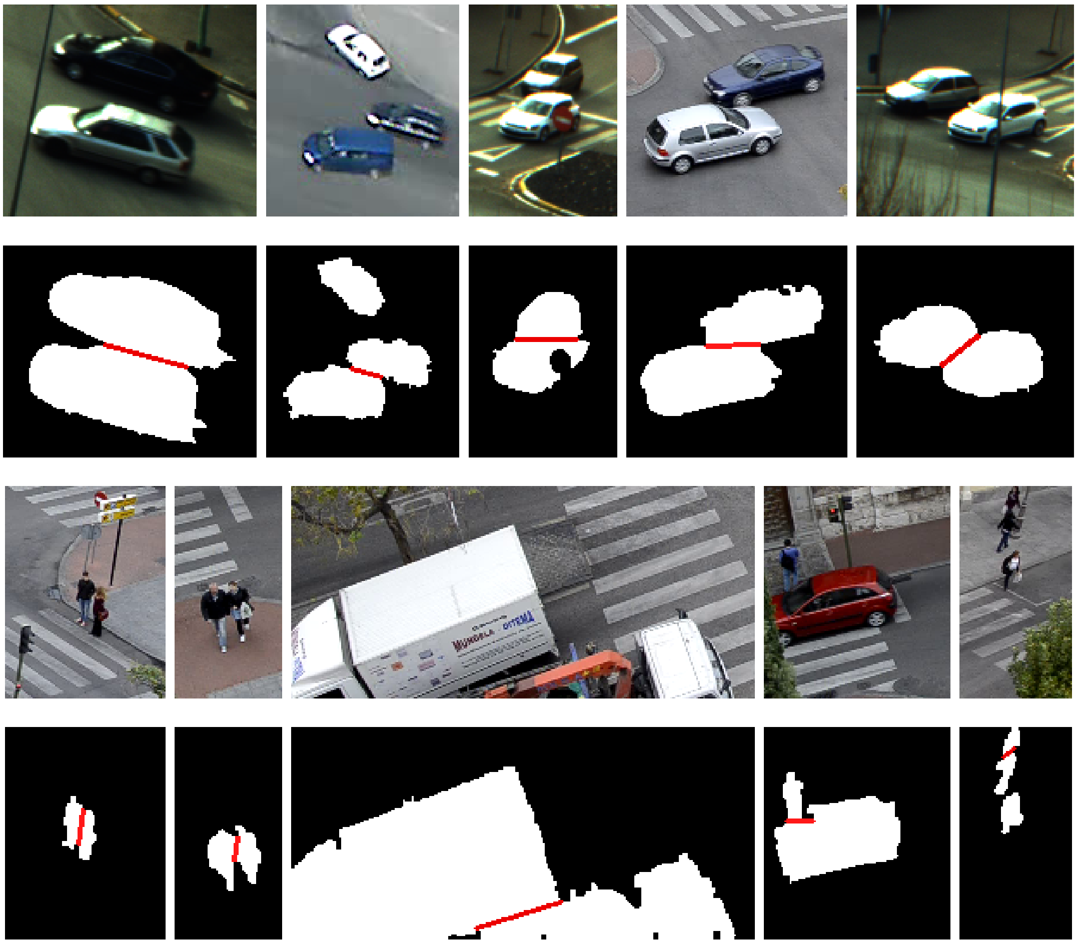


Figure 4.18: Examples of occlusion management by the proposed algorithm

The algorithm is run multiple times for each frame through the whole image until the number of blobs remains constant. Therefore occlusions with more than two objects involved can also be handled as can be seen in Figure 4.19, with 3 cars and 3 pedestrians.

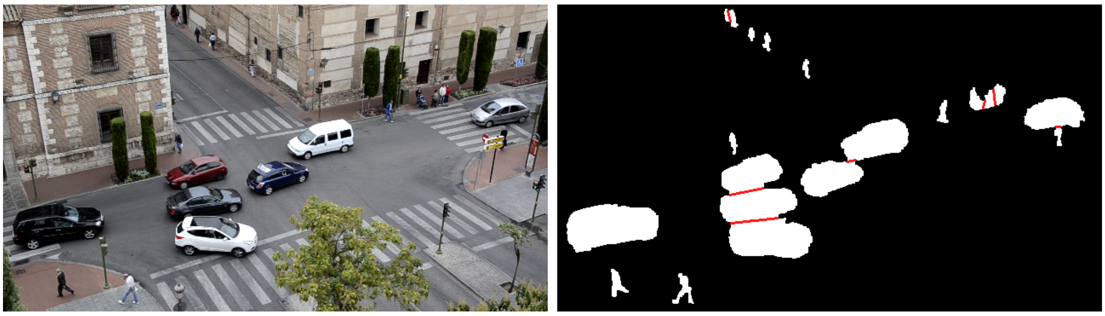


Figure 4.19: Example of occlusion management of multiple objects.

4.4.2 Feature clustering

To group all the features from the same object, a 2-stage 3D clustering algorithm is used. First the individual features are associated to a blob (after the occlusion reasoning) and grouped into ”small” clusters depending on their motion. Finally these clusters are grouped into object-level ones depending on the 3D sizes and motion.

Blob clustering

In case of the previous algorithm fails, and an occlusion is not correctly handled, there is another chance to separate different objects by clustering the features based on their motion. Therefore, if a blob corresponds to a single object, all its features will have a similar RGB motion component and will be grouped together. Otherwise, the features will be clustered into multiple objects associated to different motion characteristics. As an unsupervised stage, it is necessary to identify the number of clusters and the correspondence of the samples automatically. Hence, Mean Shift [49] is used as a non-parametric method which does not require prior knowledge of the number of clusters, and does not constrain their shape.

The main idea behind mean shift is to treat the points in the d-dimensional feature space as an empirical probability density function where dense regions in the feature space correspond to the local maxima or modes of the underlying distribution. For each data point in the feature space, one performs a gradient ascent procedure on the local estimated density until convergence. The stationary points of this procedure represent the modes of the distribution. Furthermore, the data points associated with the same stationary point are considered members of the same cluster. The quality of the output is controlled only by a kernel *bandwidth*, and it is not critical due to objects moving with different angles or velocities generate RGB features with a strong different component. Figure 4.20 depicts the clustering result of the features extracted in Figure 4.11.

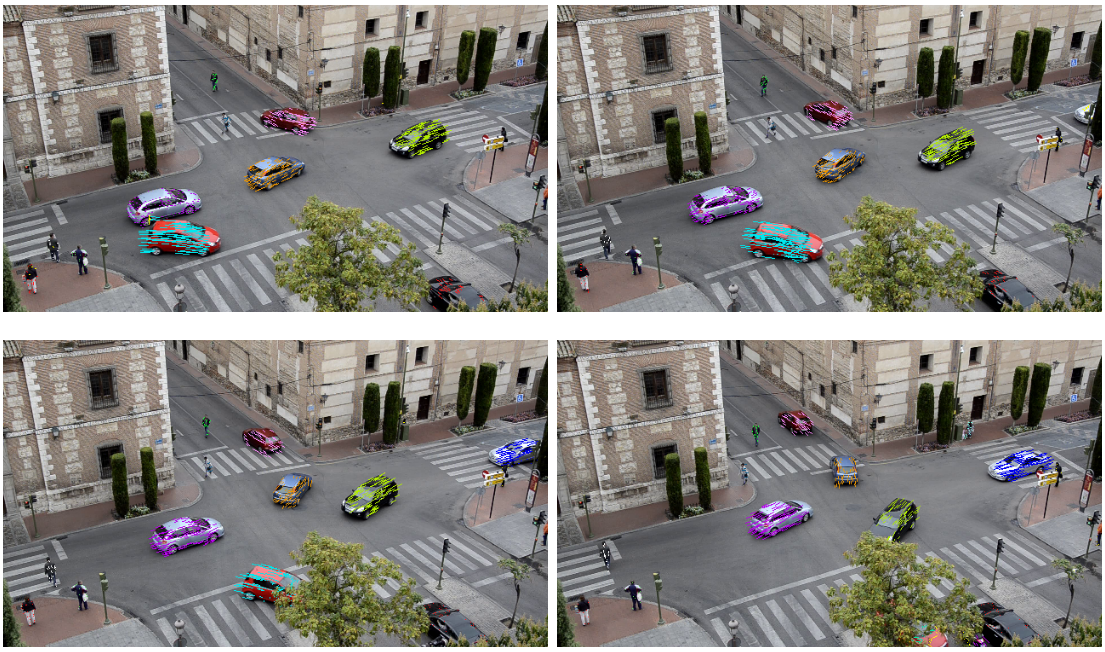
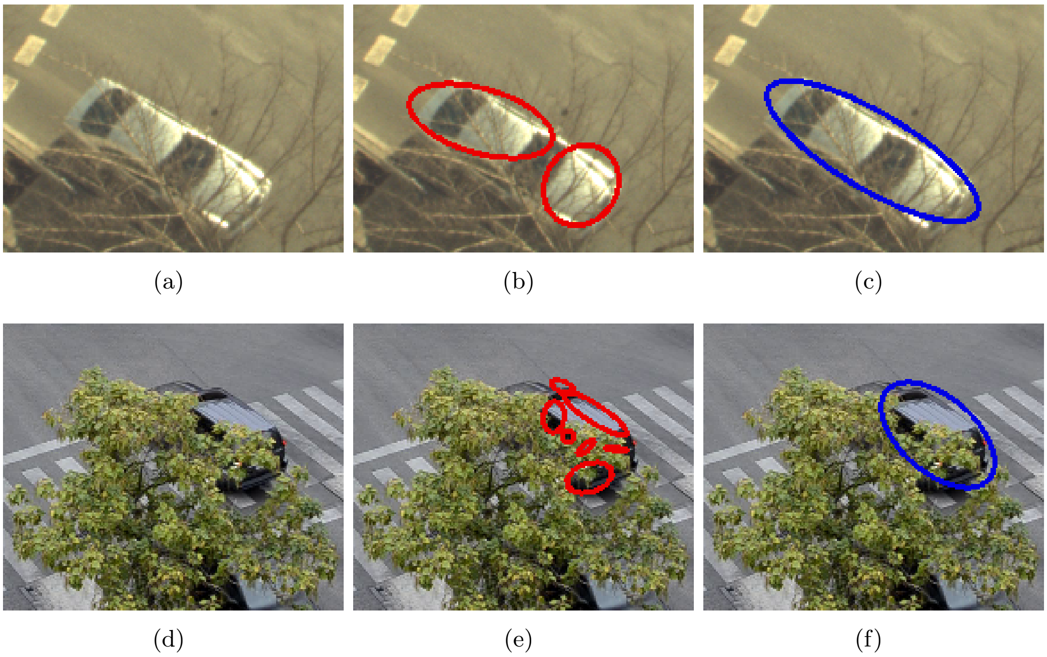


Figure 4.20: Examples of feature clustering represented by coloured features.

3D model fitting

As mentioned before, an approximate size of vehicles is known thanks to the information provided by the camera calibration. Therefore, a vehicle which has been split into several blobs due to errors in the foreground extraction or a misclassified occlusion can be merged. If the clusters fits into the 3D size of an standard vehicle in the corresponding 2D coordinates and have similar motion, the clusters are merged into a final object, represented by an ellipse. Figure 4.21 shows an example of blob merging after splitting the initial blob due to an occlusion with a tree.



(a) (b) (c)

(d) (e) (f)

Figure 4.21: Examples of cluster merging. (a-d) Source image. (b-e) Blob feature clustering. (c-f) Cluster merging by 3D model fitting.

4.5 Tracking

After detecting consecutively a cluster several times, a tracking stage combined with a multi-frame validation process takes place. This final step is used to reinforce the coherence of the detected objects over time, obtaining a more stable position, avoiding occlusions in case the previous methods fail, and minimizing the effect of both false-positive and false-negative detections. The multi-frame validation and tracking algorithm relies on the Kalman filter theory in 2-D space (image plane). For this purpose, a dynamic state model is defined considering the following state vector:

$$s_i^k = \{cx_i^k, cy_i^k, w1_i^k, w2_i^k, \alpha_i^k, \dot{cx}_i^k, \dot{cy}_i^k, \dot{w1}_i^k, \dot{w2}_i^k, \dot{\alpha}_i^k\}^T \tag{4.17}$$

where i and k correspond to the instant and number of candidate respectively, cx and cy are the respective horizontal and vertical image coordinates for the centroid of the cluster, $w1$ and $w2$ are the respective major and minor axis of the cluster ellipse, and α is the motion angle. Moreover, the velocity change of the previous parameters is included as $(\dot{cx}_i^k, \dot{cy}_i^k, \dot{w1}_i^k, \dot{w2}_i^k, \dot{\alpha}_i^k)$ to facilitate the prediction of the object ellipse in the next frame. The model used to set up the transition between states is defined as:

$$\begin{bmatrix} cx_i^k \\ cy_i^k \\ w1_i^k \\ w2_i^k \\ \alpha_i^k \\ \dot{cx}_i^k \\ \dot{cy}_i^k \\ \dot{w1}_i^k \\ \dot{w2}_i^k \\ \dot{\alpha}_i^k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \Delta_t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \Delta_t & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta_t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta_t & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta_t \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} cx_{i-1}^k \\ cy_{i-1}^k \\ w1_{i-1}^k \\ w2_{i-1}^k \\ \alpha_{i-1}^k \\ \dot{cx}_{i-1}^k \\ \dot{cy}_{i-1}^k \\ \dot{w1}_{i-1}^k \\ \dot{w2}_{i-1}^k \\ \dot{\alpha}_{i-1}^k \end{bmatrix} + \vec{n}_s \tag{4.18}$$

where Δ_t is the sampling time, needed to predict the position, size and angle of the ellipses, and $\vec{n}_s$ is the noise vector associate to the system dynamics. The prediction stage of the filter is used to extrapolate the position of the objects in a new frame based on a constant velocity constrain. The prediction can be associated with new measurements or can be used to trigger detectors. A correction step uses the detection as measurement and updates the filter state with the following measurement vector:

$$m_i^k = \{cx_i^k, cy_i^k, w1_i^k, w2_i^k, \alpha_i^k\}^T \tag{4.19}$$

Finally, Equation 4.20 represents the expression which links the measurements with the system state, where $\vec{n}_m$ is the noise vector associate to the measures. It is assumed that the random variables which describe the mentioned noises are independent and with normal distributions ($\vec{n}_s \sim N(0,Q)$ and $\vec{n}_m \sim N(0,R)$).

$$\begin{bmatrix} cx_i^k \\ cy_i^k \\ w1_i^k \\ w2_i^k \\ \alpha_i^k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} cx_{i-1}^k \\ cy_{i-1}^k \\ w1_{i-1}^k \\ w2_{i-1}^k \\ \alpha_{i-1}^k \\ \dot{cx}_{i-1}^k \\ \dot{cy}_{i-1}^k \\ \dot{w1}_{i-1}^k \\ \dot{w2}_{i-1}^k \\ \dot{\alpha}_{i-1}^k \end{bmatrix} + \vec{n}_m \tag{4.20}$$

After explaining the dynamic model used for the filter, the next step is to manage the problem of the data association, one of the main issues to solve in tracking methods. Nevertheless in this work, the developed feature tracking is very useful for this purpose. The diagram of Figure 4.22 illustrates the proposed idea. The motion vectors in the current frame (black lines) contain information of the current ellipse (red) and also of the ellipse of the previous frame (blue). Therefore the association of ellipses between frames is intrinsic of the feature tracking process. In case of some features are grouped into a different cluster, or came from different clusters, the vote of the majority is used for the data association.

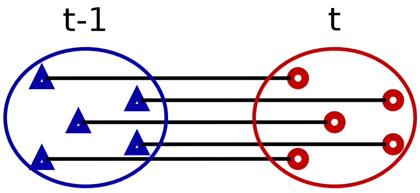


Figure 4.22: Diagram of data association for tracking.

Figure 4.23 depicts a sequence of images where tracking reveals fundamental. Due to an occlusion between a vehicle and a pedestrian, a very small part of the last one is visible and is not detected as an occlusion. However, as the objects were previously taken into tracking, they are kept separated.

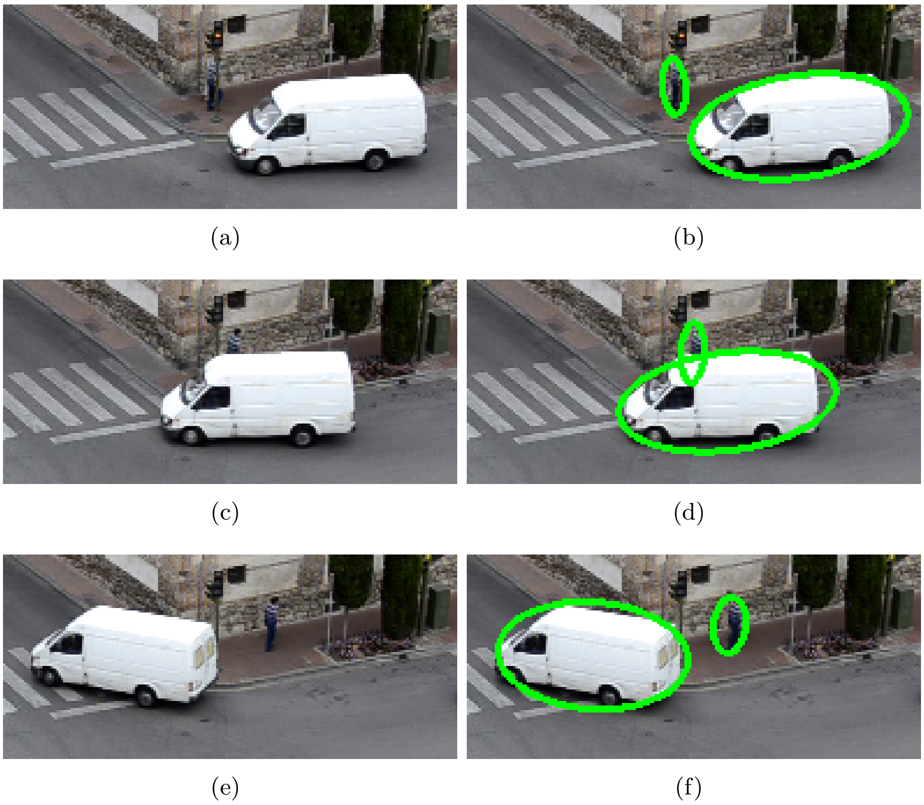

(a) (b) (c) (d) (e) (f)

Figure 4.23: Example of object tracking advantages. (a-b) before the occlusion. (c-d) during the occlusion. (e-f) after the occlusion.

At the same time, the tracking stage is able to manage two common problems of background subtraction: the *ghosts effect* and the *stationary objects*.

The ghosts problem is associated to background regions misclassified as a foreground object. This is due to a sudden change of the scene, that differs from the modelled background, mainly produced by stopped objects included in the background model which change their position. Therefore two new objects are extracted, the moving object and the area where it was located and totally unknown for the GMM. The solution is based on the features motion history and the ellipse tracking as follows: if a new object is created and its position does not change from the initial one, the object is reconsidered as background. The effect and the solution are illustrated in Figure 4.24. Since the beginning of the video the vehicle was stopped in the initial position. Once it changes its position, it is detected as foreground and tracked, but not as well as the area where it was placed, although it is also considered as foreground. After the updating time, the ghost is absorbed by the background model.

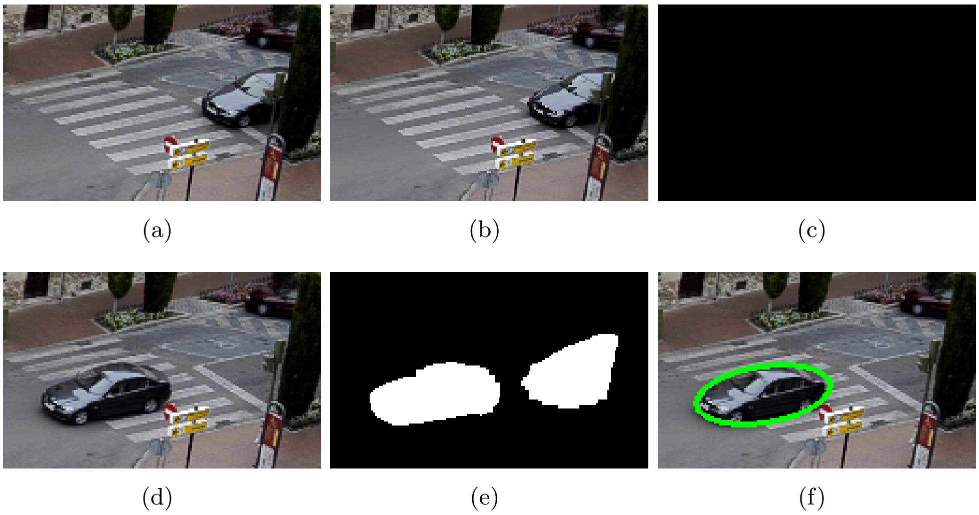


Figure 4.24: Example of a ghost management. (a) Source image before ghost effect. (b) Background model. (c) Foreground image before ghost effect. (d) Source image during ghost effect. (e) Foreground image during ghost effect. (f) Detected object with ghost managed.

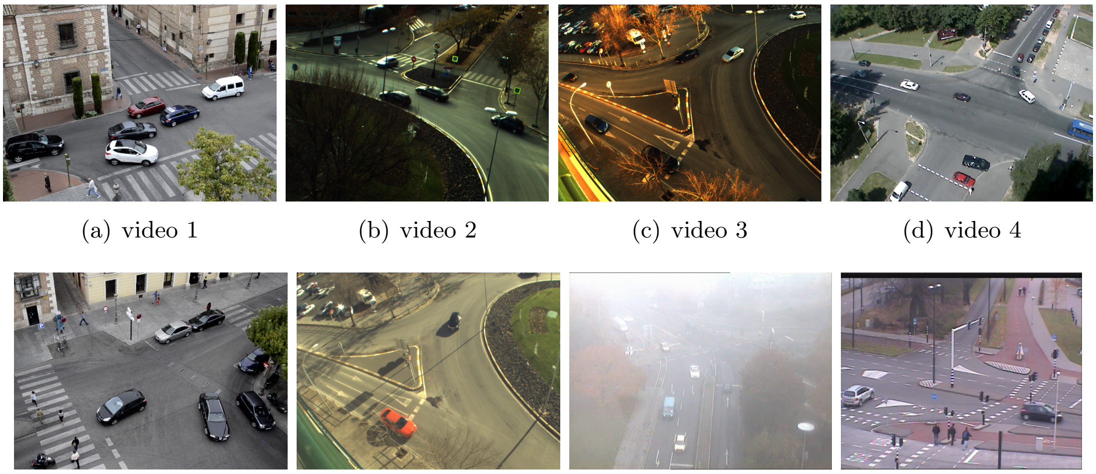
On the other hand, stationary objects are moving objects considered as foreground that stop and remain quiet longer than the background updating rate. Hence, they are absorbed by the model as background. Tracking stage keeps them into analysis during a certain time, until considering the object part of the background like a parked vehicle.

4.6 Experimental results

To analyse the performance of the developed approach, the algorithm has been tested on over 2 hours of traffic videos with more than 2000 objects between vehicles and pedestrians. The sequences include different camera views, illumination effects, shadows, etc., in order to evaluate the method in a wide range of situations. Some examples of the testing scenarios used are shown in Figure 4.25 and described in Table 4.3.

Video	# frames	Resolution	Conditions	Source
video1	16402	640x480	Cloudy	Own sequence
video2	5244	640x480	Dusk (dark)	Own sequence
video3	3332	640x480	Dusk (bright)	Own sequence
video4	18296	640x480	Sunny	Lunds Univ. [50]
video5	15921	640x480	Cloudy	Own sequence
video6	3585	640x480	Sunny	Own sequence
video7	630	768x576	Fog/rain	Karlsruhe Univ. [51]
video8	4290	352x288	Cloudy	Candela [52]

Table 4.3: Description of testing videos.



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

(e) video 5 (f) video 6 (g) video 7 (h) video 8

Figure 4.25: Samples of testing scenarios.

Firstly, as a complement of the study of the object occlusion reasoning described in Subsection 4.4.1, the performance of the proposed framework has been quantitatively evaluated on the analysed sequences. From a total of 532 occlusions, the results are summarized in Table 4.4, separated by occlusion class (pedestrian-to-pedestrian, car-to-car, car-to-pedestrian or full occlusion) and depending on the level of the algorithm where they were detected and managed (occlusion level, clustering level or tracking level). *Detected* columns stand for the number of occlusions detected by each level, and *handled* is the number of occlusions correctly managed by each level. Occlusion level always takes part in the process and only if it can not detect or handle the occlusion, the algorithm passes through the next level. Because of that, the numbers of the clustering and tracking levels are smaller.

	Occlusion level		Clustering level		Tracking level		Together			
	Detected	Handled	Detected	Handled	Detected	Handled	Detected	Handled	Total	Rate
Ped&Ped	226	213	11	11	18	15	255	239	267	0.89
Car&Car	124	115	19	18	5	4	148	137	147	0.93
Car&Ped	53	51	29	26	9	8	91	85	94	0.90
Full	0	0	0	0	22	22	22	22	24	0.91
Result:	**403**	**379**	**59**	**55**	**54**	**49**	**516**	**483**	**532**	**0.91**

Table 4.4: Quantitative evaluation of the occlusion reasoning framework.

From the results represented in the table, several conclusions can be extracted:

- Its lower ratio shows that pedestrian-to-pedestrian occlusions are the most problematic ones. This result was expected due to the small size and motion variability of the objects. On the other hand, and for the opposite reasons car-to-car occlusions have the highest ratio.

- Car-to-pedestrian occlusions generate a small area an convexity hull that sometimes does not fit with the requirements of the occlusion reasoning algorithm. Because of that this ”detected” value is so low. However the rest of levels can deal with the situation and are able to manage the occlusion.

- As a single frame analysis, full occlusions can not be detected by the two first levels. Only a multi frame algorithm as the tracking one can handle them.
- The global occlusion management ratio (91.4%) is very reasonable. It is important to emphasize that this analysis is single frame. Therefore an error due to an occlusion in a particular frame is not important in the whole path of an object. The advantage of the system is the use of a multi-level framework that allows to solve an occlusion from 3 different and complementary points of view.

Next, the global results of the application are depicted in terms of object detection rate, recall and precision. The *Detection Rate* (DR) is the percentage of correctly detected objects, the *Recall* (R) measures the system's ability to identify positive samples, and the *Precision* (P) is the fraction of retrieved instances that are relevant. These two last indicators are defined as:

$$Recall = \frac{TP}{TP + FN} \tag{4.21}$$

$$Precision = \frac{TP}{TP + FP} \tag{4.22}$$

where TP stands for the number of true positives (objects correctly detected at least the 80% of their path), FP stands for the number of false positives (unexpected detections or object splits) and FN is the number of false negatives (missing detections). These parameters are manually extracted with the final tracking results. In order to join all indicators into one, the *F-measure* (F) is defined as a measure of the test's accuracy by:

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.23}$$

For a better understanding and comparison of the results, the mentioned indicators have been computed for each object class (pedestrian, car, van, etc.) and each sequence class (sunny, cloudy, rainy, etc.), and divided into two tables (Table 4.5 and 4.6).

Object class	N	TP	FP	FN	DR	R	P	F
Car	1105	1081	91	24	0.978	0.978	0.922	0.949
Pedestrian	877	801	5	76	0.913	0.913	0.994	0.952
Bicycle	25	23	0	2	0.920	–	–	–
Motorbike	17	16	0	1	0.941	–	–	–
Van	149	134	0	0	0.899	–	–	–
Bus	53	28	0	0	0.528	–	–	–
Truck	43	33	0	0	0.767	–	–	–
Total	**2269**	**2116**	**96**	**103**	**0.933**	**0.954**	**0.957**	**0.955**

Table 4.5: Results obtained by object class. **N**: number of samples. **DR**: detection rate. **TP**: number of true positives. **FP**: number of false positives. **FN**: number of false negatives. **R**: recall. **P**: precision. **F**: F-measure.

From a total amount of 2269 objects, the system has obtained a detection rate of 93.3%. However, this value can be considered higher if the following considerations are taken into account:

- Pedestrians usually have a very small image size. Because of that, pedestrian-to-pedestrian occlusions can not be always managed by the system, and a small group of pedestrians (commonly 2, rarely 3 or more) sometimes are considered as a single pedestrian. This issue decreases the DR and increases the FN, but is not considered crucial, because at least the system detects one pedestrian. R, P and F are affected by the same reason. Figure 4.26 depicts an example of this situation.

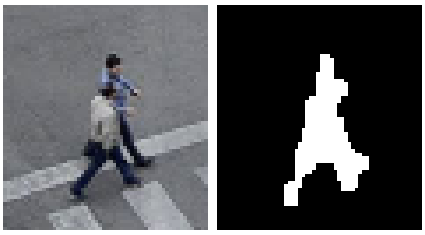


Figure 4.26: Example of non detected occlusion by two pedestrians.

- Bicycles and motorbikes produce a detection rate of 92% and 94.1% respectively. However only 3 of these objects were missing, again due to an occlusion with a car with similar motion. Similarly, for a counting system, considering one object instead two is a mistake, and the DR is decreased, but the error is acceptable. An example is shown in Figure 4.27.

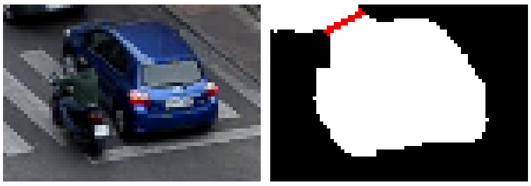


Figure 4.27: Example of non detected occlusion by a car and motorbike.

- The low detection rate obtained for vans, buses and trucks, is caused by their size. Comparing an standard 3D size of a car with a bigger vehicle can derive into blob splitting if the occlusion reasoning detects any substantial convexity defect. Therefore it does not mean they are not detected; they are detected twice and it is considered a detection error. The good point is that they are the less common vehicles. Nevertheless this is a future line to improve the system. Figure 4.28 represents an example of correct and wrong detection cases due to the mentioned effect.

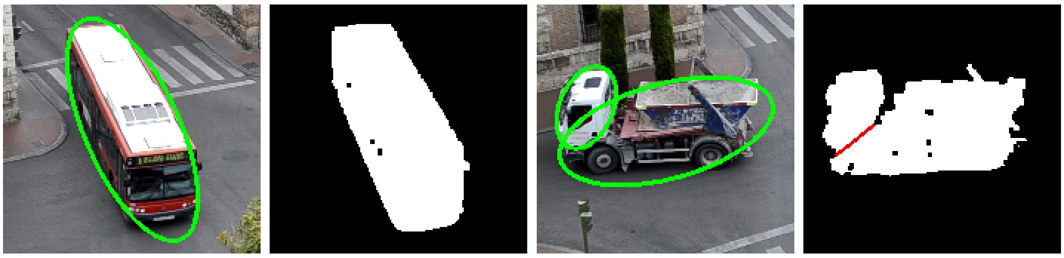


Figure 4.28: Example of correct and wrong detection with big vehicles.

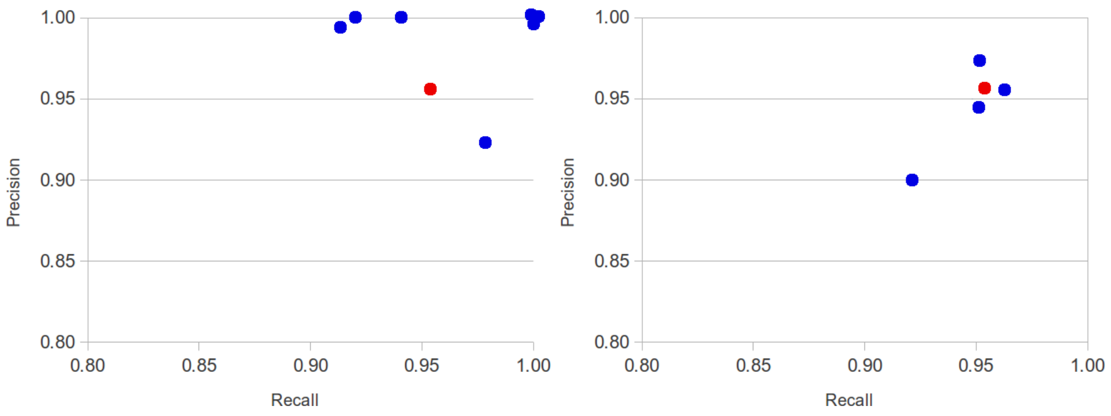
- There are no false negatives in vans, buses and trucks rows. This effect is caused because all of them are always detected, although incorrectly due to the reason explained in the previous paragraph. Consequently, the value of false positives index in the case of cars is high, and the value on R, P, and F rates is unitary for vans, buses, and trucks.

- The good detection rate is supported by the rest of high obtained ratios, all over 95%.

Scenario	N	TP	FP	FN	DR	R	P	F
Sunny (shadows)	901	832	39	32	0.923	0.963	0.955	0.959
Cloudy	885	841	23	43	0.950	0.951	0.973	0.962
Dusk	312	291	17	15	0.933	0.951	0.945	0.948
Rain/snow	171	152	17	13	0.889	0.921	0.899	0.910
Total	**2269**	**2116**	**96**	**103**	**0.933**	**0.954**	**0.957**	**0.955**

Table 4.6: Results obtained by scenario. **N**: number of samples. **DR**: detection rate. **TP**: number of true positives. **FP**: number of false positives. **FN**: number of false negatives. **R**: recall. **P**: precision. **F**: F-measure.

The results obtained by scenario show a more stable detection rate, what means that the system is more sensitive to the object type than the sequence conditions. This effect is represented in Figure 4.29, where the dispersion of values around the global one is smaller in the sequence-type classification. Special interest has the ability of the system to reliably detect vehicles with adverse weather conditions, with a detection rate of 88.9%. The system is also able to work with different types of perspectives, since it computes the calibration of the camera and thus considers the 3D volume of vehicles instead of just 2D silhouettes. Therefore the scenario characteristics are not crucial for the system, and the single ratio values are not representative by scenario.



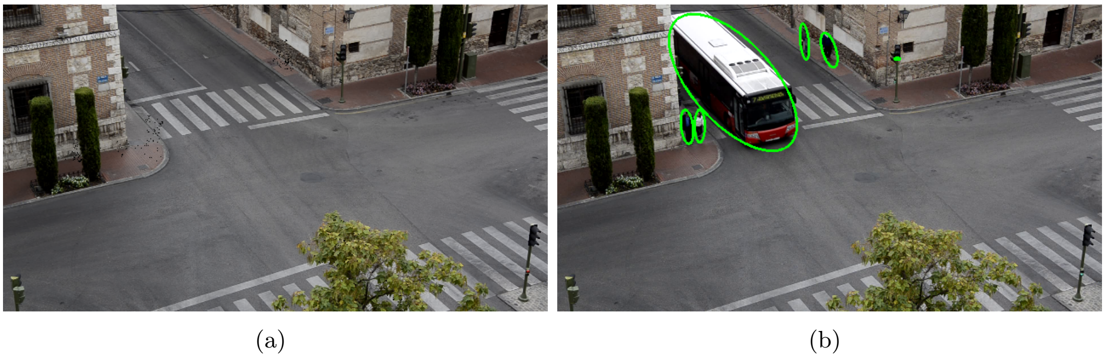
(a) By object class

(b) By scenario

Figure 4.29: Recall and precision graphs for the tested sequences represented in Tables 4.5 and 4.6. Blue dots are associated to the single values obtained and Red dots represent the total values.

To conclude this section, some results are graphically depicted from the sequences described previously.

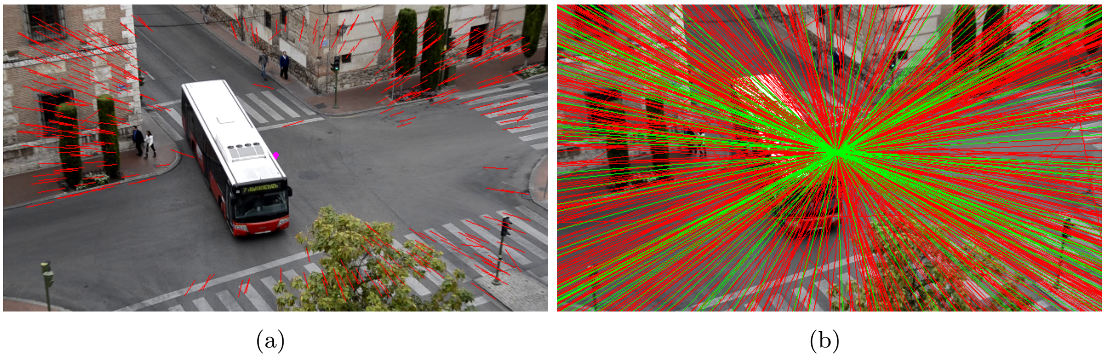
The first sequence shows a zoom change and how the system behaves before, during and after this process. Figure 4.30 represents the background model before the zoom change and shows some object detections.



(a) (b)

Figure 4.30: Sequence 1 before zoom change. (a) Background model. (b) Object detection.

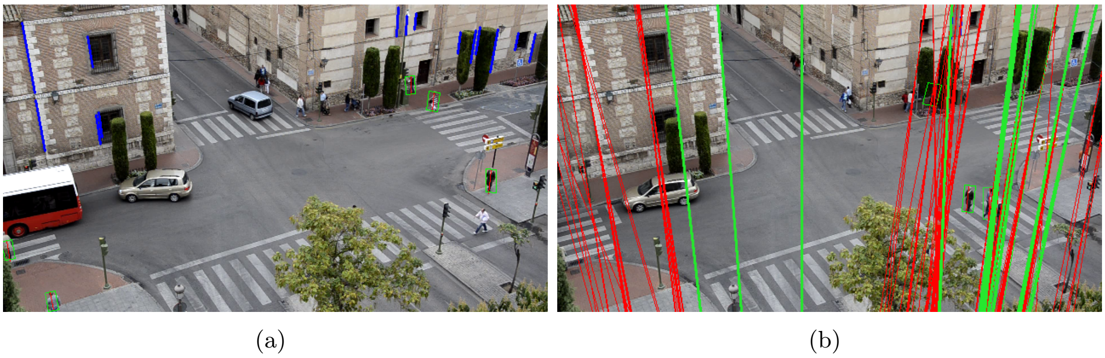
Suddenly a camera zoom is detected by the image stabilization module so the background model is restarted and the principal point computed (Figure 4.31).



(a) (b)

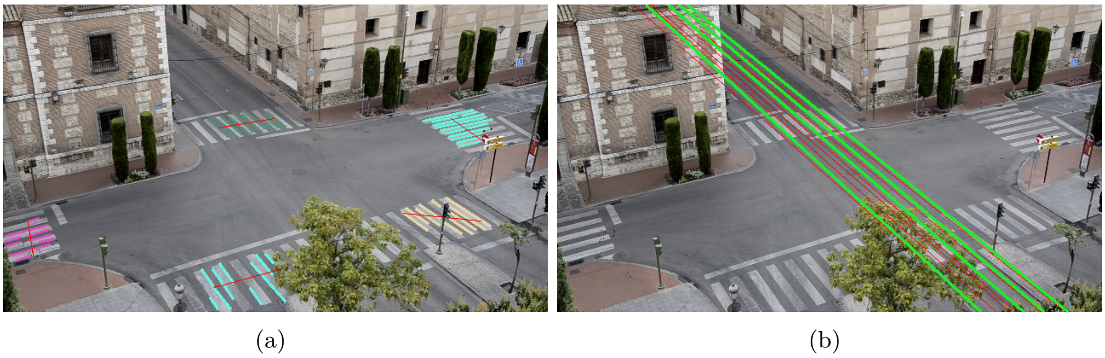
Figure 4.31: Sequence 1 during zoom change. (a) Image stabilization detects zoom change. (b) Principal point detection.

Next, as a zoom change, two new vanishing points are needed to recalibrate the camera. Figures 4.32 and 4.33 depict the search of these vanishing points.



(a) (b)

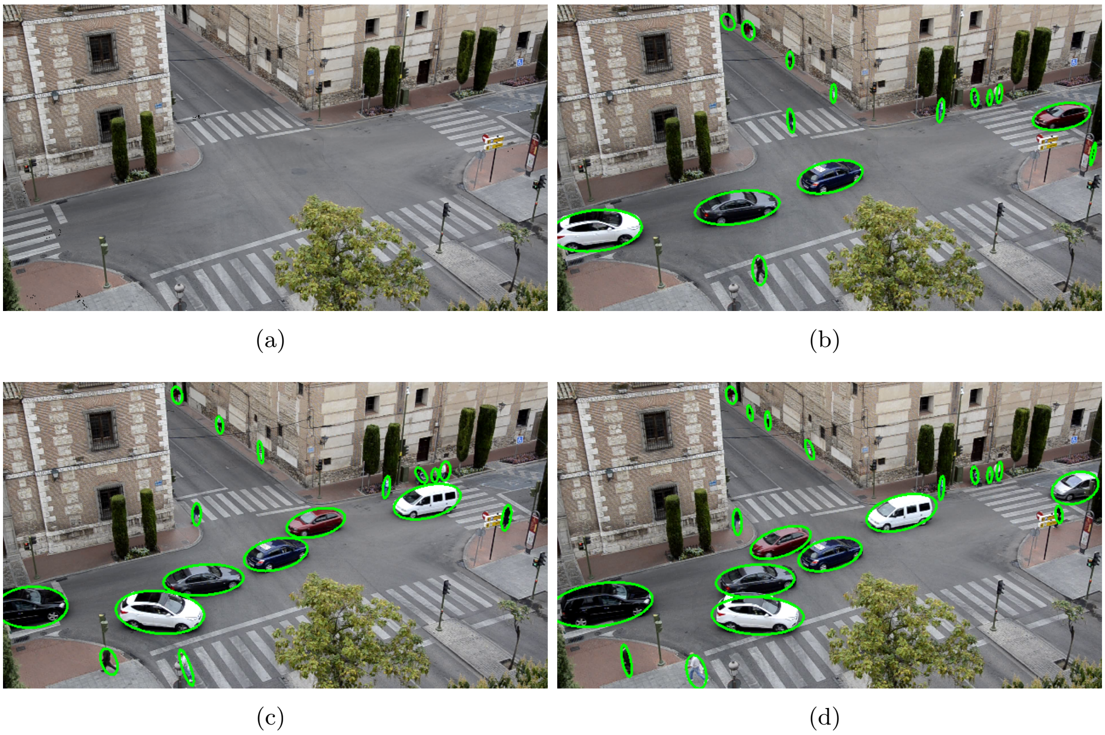
Figure 4.32: Sequence 1 after zoom change. (a) Searching vertical lines to compute the vertical vanishing point. (b) Vertical vanishing point extraction.


(a)

(b)

Figure 4.33: Sequence 1 after zoom change. (a) Searching crosswalk parallel lines to compute the ground plane vanishing point. (b) Ground plane vanishing point extraction.

Finally, the new background model is recomputed, as shown in Figure 4.34 and new detections are performed.


(a)

(b)

(c)

(d)

Figure 4.34: Sequence 1 after zoom change. (a) New background model. (b-d) Object detection.

The next pages represent some samples of graphical results for other 7 sequences. As can be seen, they provide different illumination conditions, camera positions, etc. to cover as many possible scenarios.

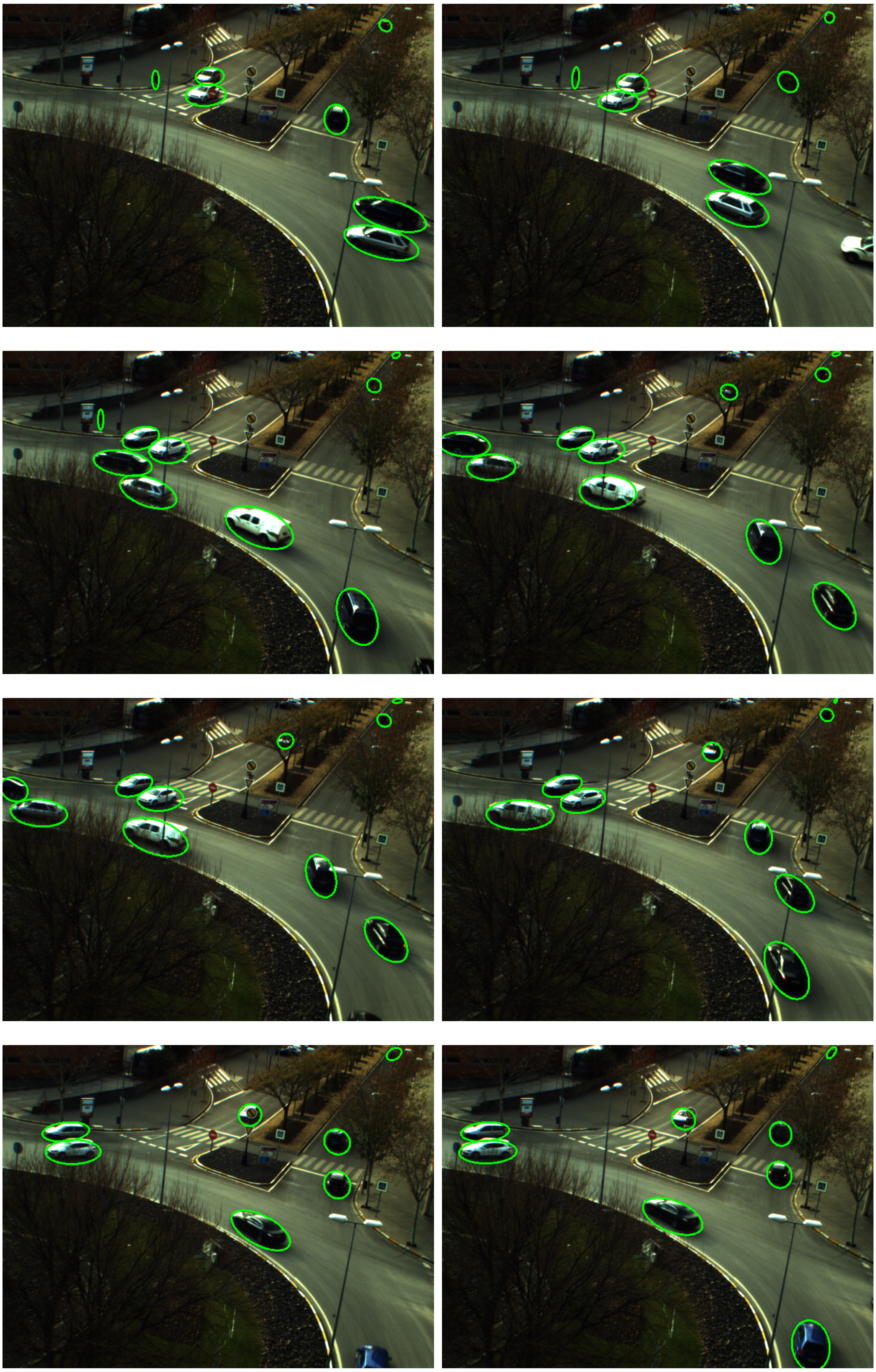


Figure 4.35: Graphic results of video 2. Samples in dusk conditions with some occlusions.

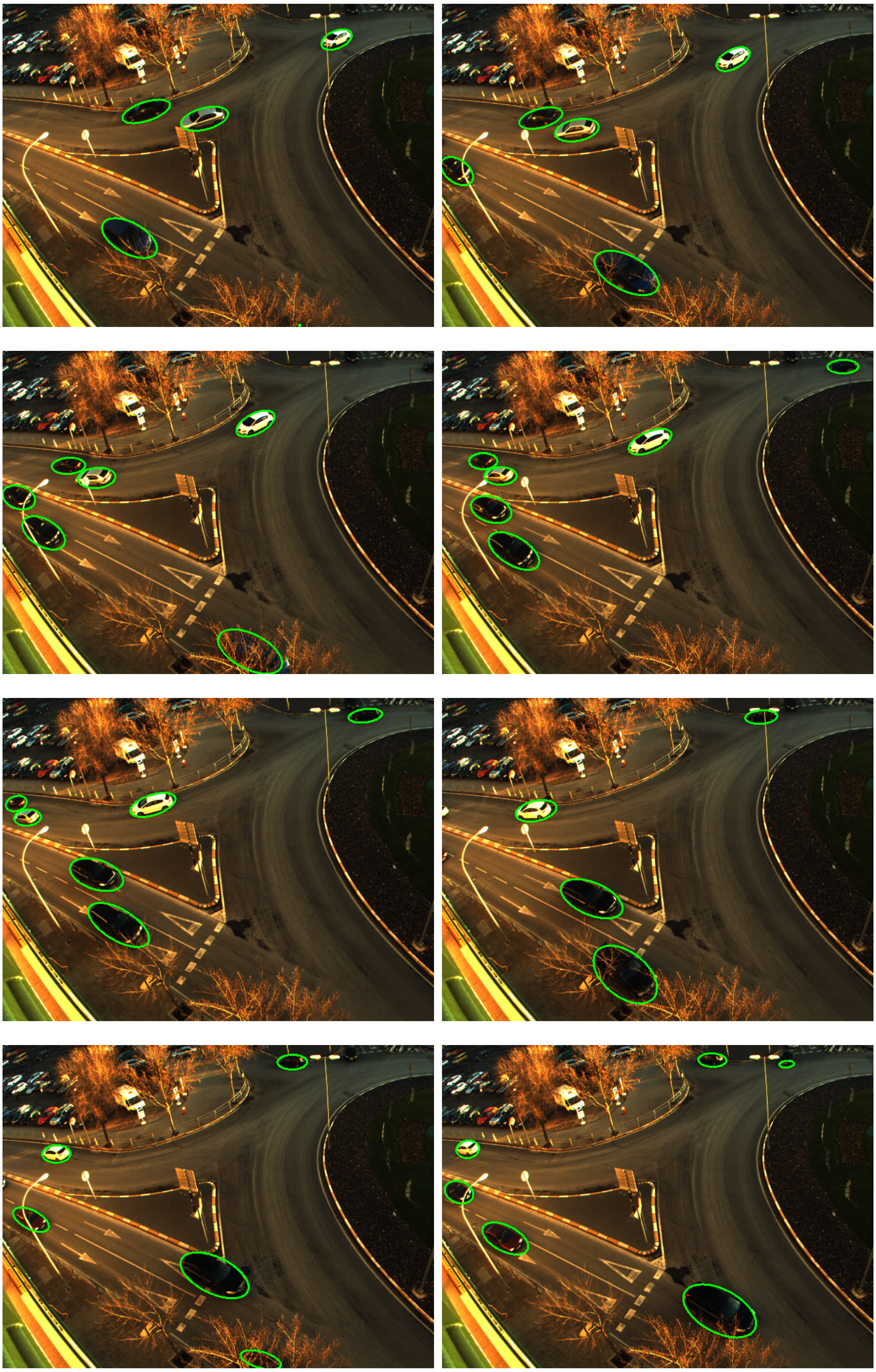


Figure 4.36: Graphic results of video 3. Samples in dusk conditions with some occlusions.

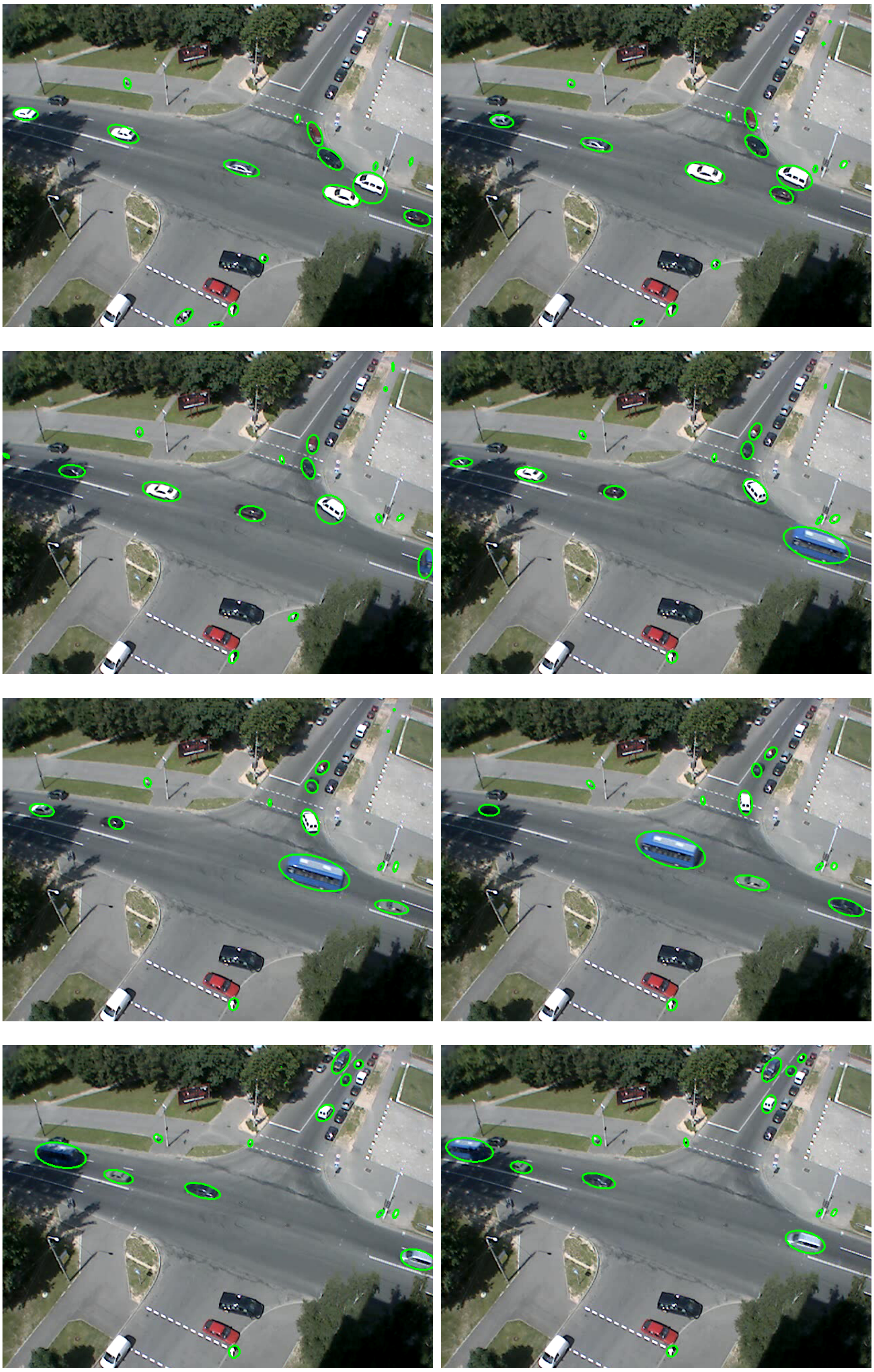


Figure 4.37: Graphic results of video 4. Samples in sunny conditions with a bus detection.

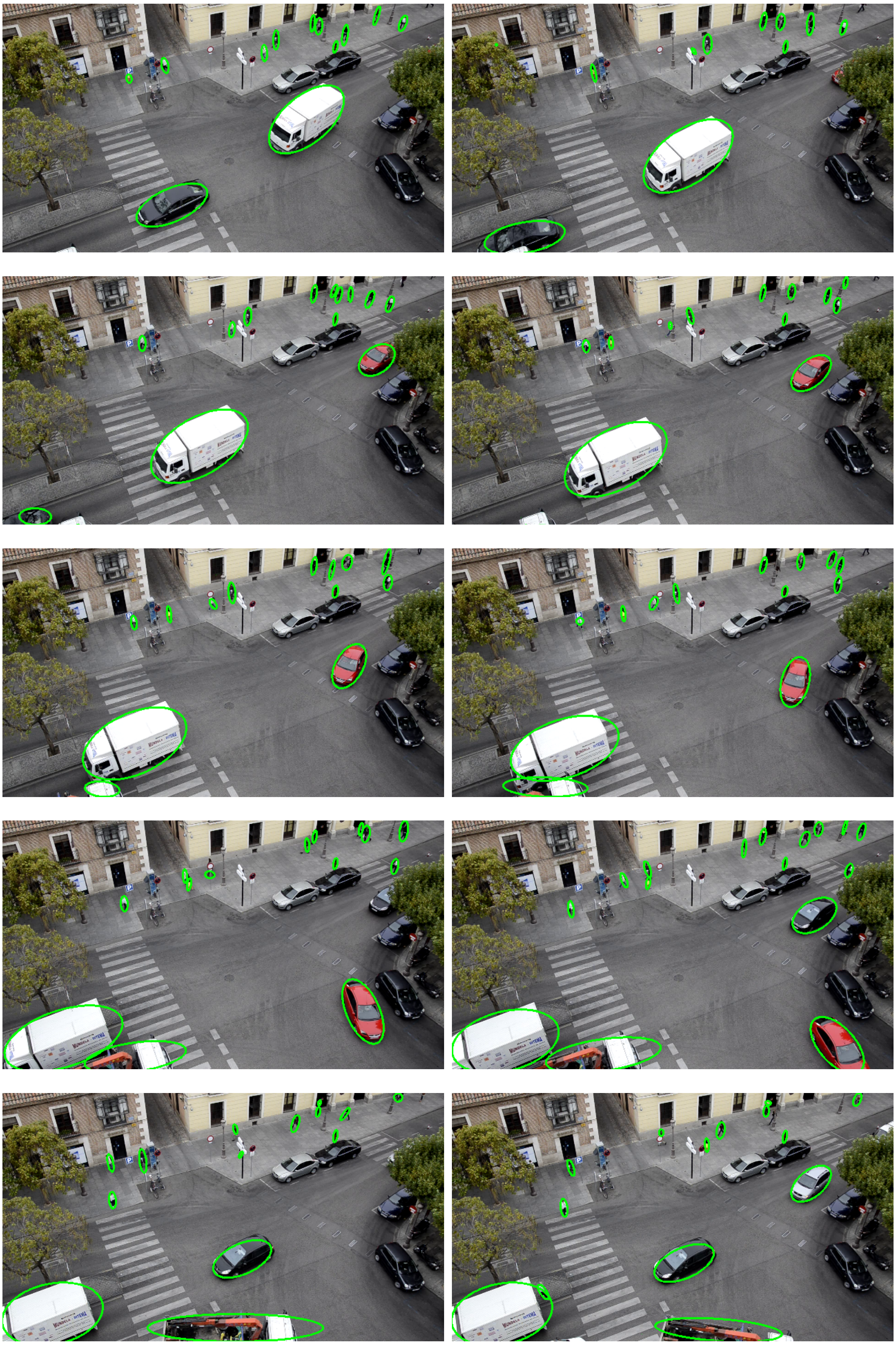


Figure 4.38: Graphic results of video 5. Samples in cloudy conditions with occlusions and a truck detection

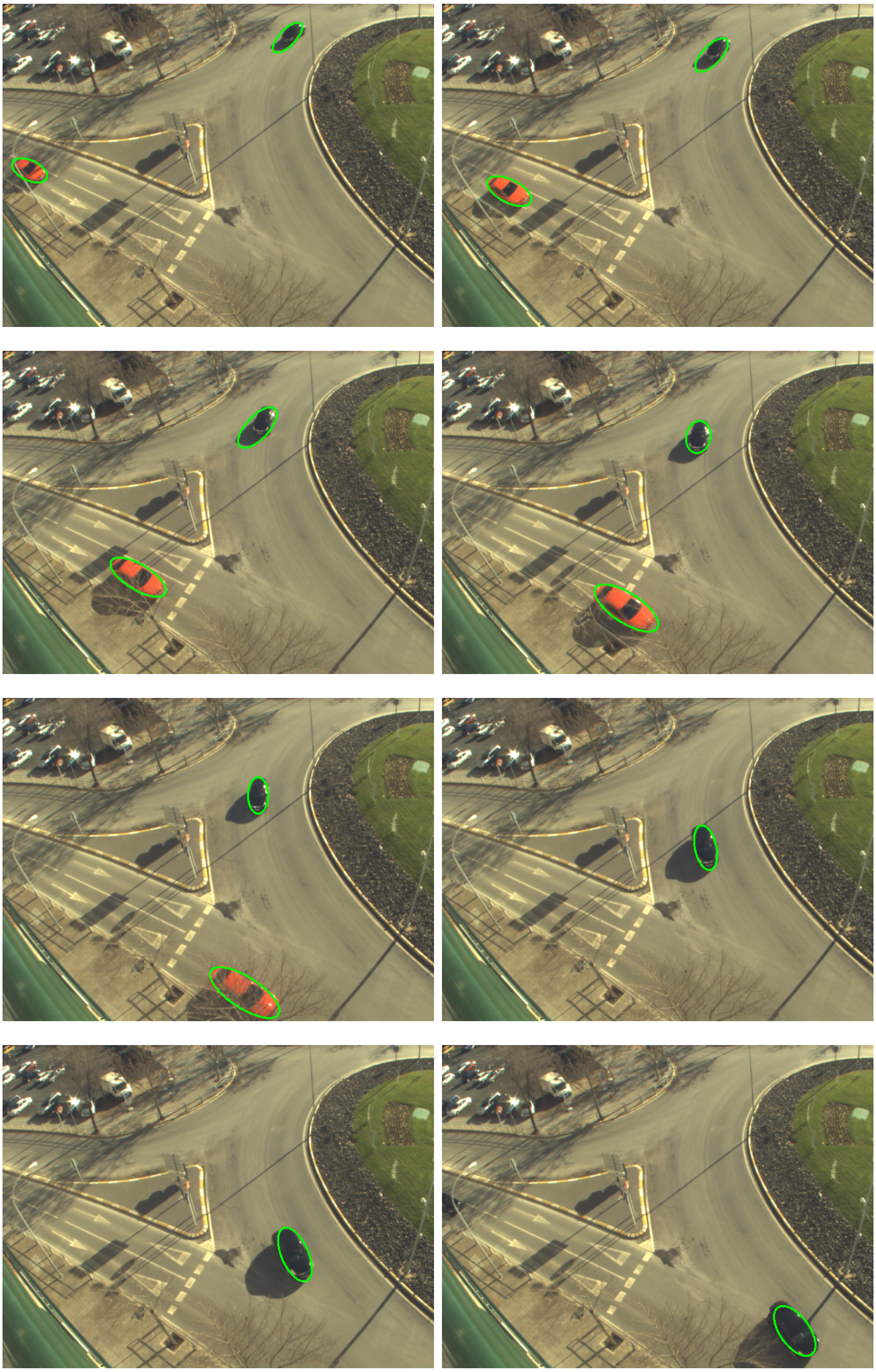


Figure 4.39: Graphic results of video 6. Samples in sunny conditions with representative shadows.

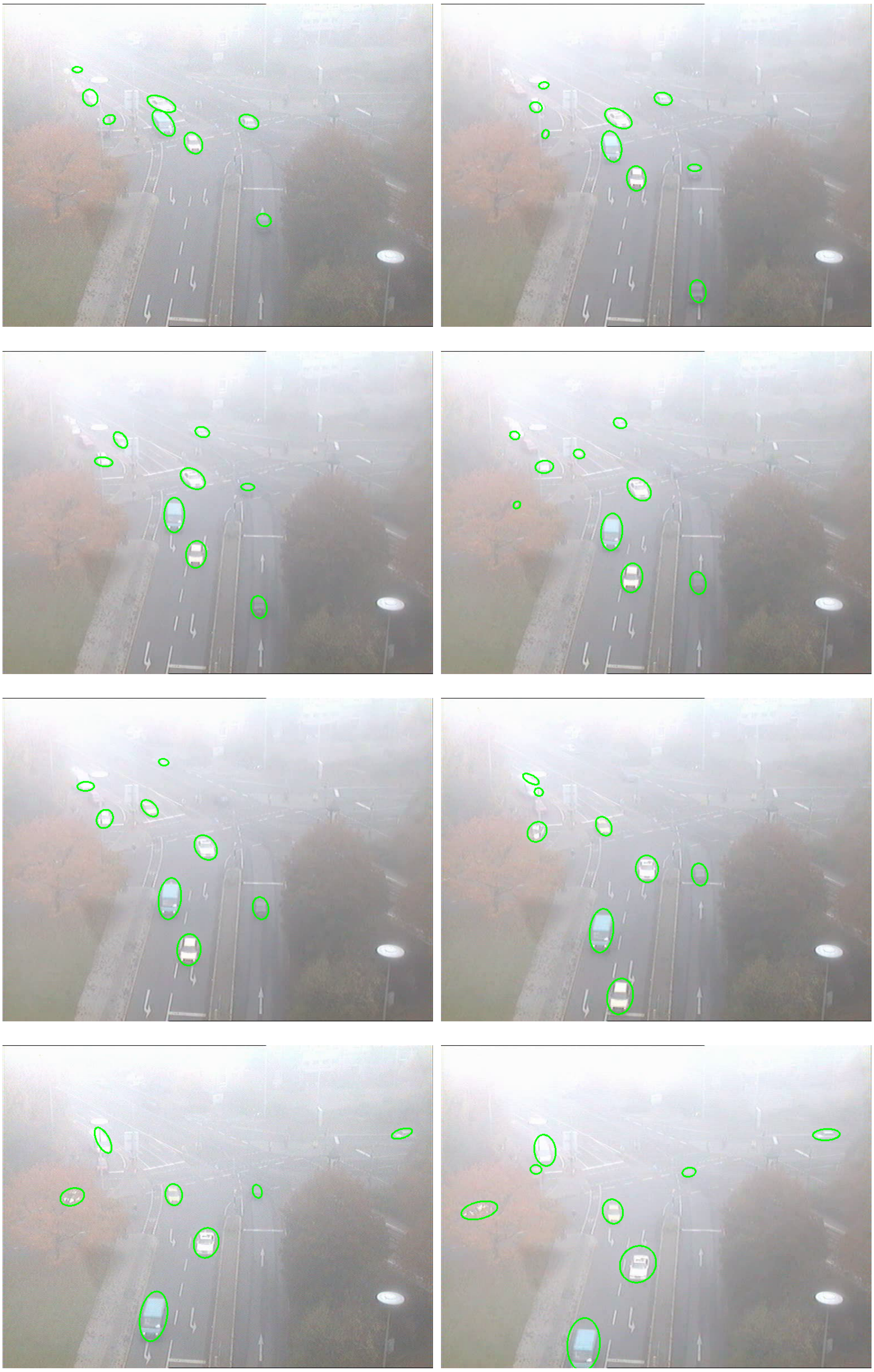


Figure 4.40: Graphic results of video 7. Samples in foggy conditions

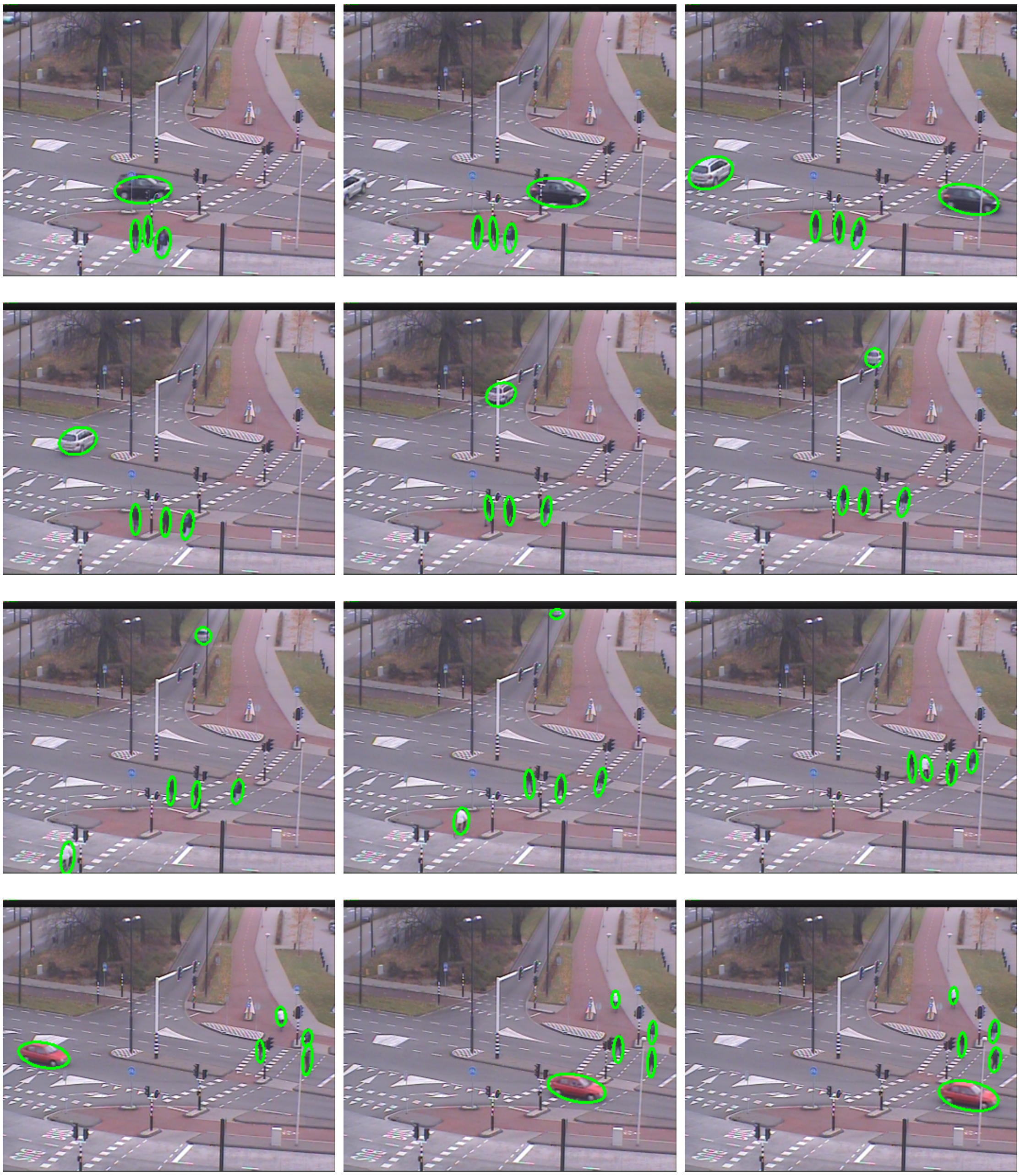


Figure 4.41: Graphic results of video 8.

4.7 Conclusions

In this chapter, a multilevel framework for target detection and tracking has been presented. Through four levels (image segmentation, feature analysis, feature clustering and object tracking) the system is able to detect and track pedestrians and vehicles with satisfactory results. The performance of the system has been described through the results obtained by analysing over 2 hours of traffic videos with more than 2000 objects.

The main problems associated to background subtraction are managed through a high level image analysis to detect camera vibrations and illumination changes.

Moreover feature analysis provides useful information about moving objects (velocity, motion direction, etc.) without any prior information or model, just two consecutive frames. It makes the motion estimation process less sensitive to different weather conditions, fragmented objects, etc. Even although it was not an objective of this thesis, as can be seen in Figure 4.42, the system has very interesting results with low artificial illumination, which give the chance to extend and improve the system for night conditions. As demonstrated in the previous section, the system is fully adaptable to the scene, so results are mostly independent of it.

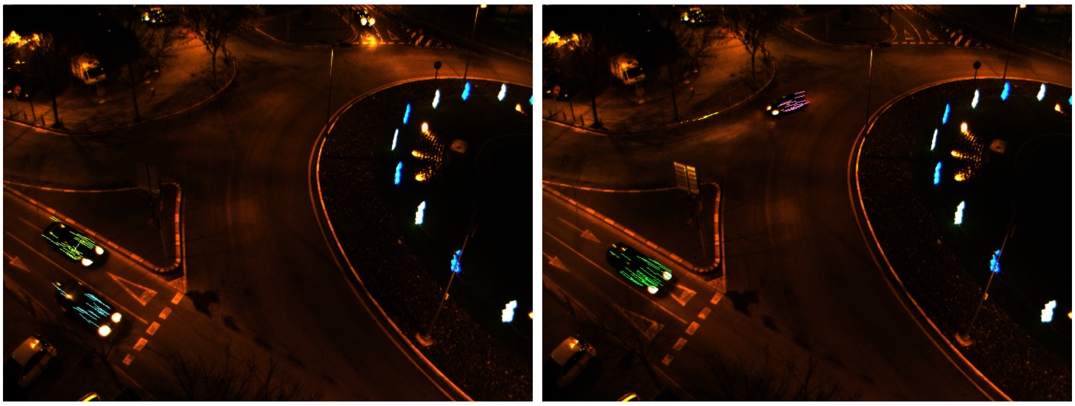


Figure 4.42: Feature extraction in night conditions.

As shown by Table 4.4 the global occlusion management ratio is very reasonable, in spite of being a single frame ratio. Furthermore, the good detection rates and indicators for the global application, and the versatility of the algorithms to multiple conditions prove that the proposed approach is a good basis for an automatic traffic surveillance system.

Chapter 5

Conclusions and future work

A number of strategies and algorithms have been devised and described in this work, with special interest in the camera auto-calibration process. The following paragraphs present the global conclusions and discuss the main contributions introduced and developed along the chapters of the thesis, pointing out the achieved enhancements, and also addressing their limitations. This discussion guides the future work section, in which potential evolutions of the presented work are summarized.

5.1 Conclusions

As the main objective of the thesis, a monocular system has been developed to detect and track vehicles and pedestrians for applications in the framework of Intelligent Transport Systems. The algorithm requires no object model or prior knowledge (only an approximate size of the searched objects in world coordinates) and it is robust to illumination changes, shadows and occlusions. Therefore it can work indoor and outdoor, in different conditions and scenarios. Moreover, due to a hierarchical camera auto-calibration process based on vanishing point extraction, the system is completely autonomous ("plug&play"), independent of the position of the camera and able to manage pan-tilt-zoom changes in fully self-adaptive mode.

Auto-calibration

A novel hierarchical self-calibration procedure based on vanishing points has been presented and discussed. Depending on which elements appear in the scene and the chance of using camera zoom, 5 levels have been established to determine the hierarchy of each developed method and the priority of the solution adopted. It is an important step for the final goal of the thesis, because it provides very useful information to compute an approximate size of the searched objects, necessary for the target detection a tracking algorithm proposed.

To test the performance of the approach, 30 sequences from different scenarios and conditions have been used. The obtained results are really satisfactory: the low error of the 3D prisms projections and distance measurements proves the strength of the system, and the multiple options of the hierarchical tree provide high versatility to cover most of the possible traffic scenarios and possible configurations without any restriction in terms of constraints or the need of prior knowledge. Furthermore, the system is able to adapt the calibration parameters in case of PTZ camera displacements without manual supervision. In case that there is no chance to auto-calibrate the camera (due to absence of orthogonal components), an interactive tool has been developed to manually input the sets of orthogonal lines, to allow the user to control the system in a short time.

Finally, the acceptable variation ranges of the vanishing points coordinates (studied in the sensitivity analysis), give the algorithm a tolerance of at least 30 pixels, which means that small errors are not critical.

Image segmentation

The proposed approach is based on the background subtraction technique. Rather than explicitly modelling the values of the pixels as one particular kind of distribution, each pixel is modelled by a mixture of K Gaussian distributions (Gaussian Mixture Model), whose mean and variance is adapted over time. The use of an adaptive method makes the system flexible to changes in the scene. Moreover the Gaussian Mixture Model allows to work with multi-layer backgrounds, where objects with repetitive movements that belongs to the background, like trees, are incorporated into the model.

The main problems associated to the background subtraction technique are managed through a high level image analysis to detect camera vibrations and illumination changes. The implemented image stabilization module neutralizes the possible camera shake and it is able to detect PTZ camera displacements to reset the background model and restart the auto-calibration if necessary. Moreover this module is used to compute the principal point of the image through camera zooming for camera calibration.

In case of cast shadows and sudden illumination changes, a module to detect and remove these foreground distortions has been implemented based on Color Normalized Cross Correlation and Gaussian Mixture Shadow Model. It works fairly good in all tested sequences, however, as a deterministic approach it probably will fail representing really strong shadows where color and chromaticity information are totally lost. It is a general problem with many proposed works but no satisfactory results, so a complete thesis could be dedicated entirely to this topic.

Feature analysis

Foreground features are extracted and tracked using FAST and KLT techniques with flock of features constraints. This methodology can provide useful information about moving objects without any prior information or model. The idea of the algorithm is to extract and track foreground features to further clustering them into objects using proximity, motion history, speed, orientation and the size constraints provided by the calibration. The algorithm can be used in daylight, twilight or night-time conditions, as well as different traffic conditions and camera positions. It is self-regulating because it selects the most salient features under the given conditions. Therefore this module also provides a strong adaptability to the scene, to make the system as independent as possible to external conditions.

Finally, the motion of the features is represented in a novel RGB-motion space in order to avoid problems with similar directions but different angles, and to facilitate the posterior clustering step.

Occlusion reasoning and clustering

One problem associated to traffic surveillance is the high probability of occlusions, due to the camera perspective, and the derived difficulties to extract the different targets of the scene. To efficiently deal with the problem, a novel multilevel clustering algorithm is presented. First an occlusion reasoning step is done in order to split foreground blobs from different objects, based on convexity defects of the occlusion blob. After that, the individual features are associated to a blob and grouped into “small” clusters depending on their motion using Mean-Shift. Finally, these clusters are grouped into object-level ones depending on the 3D sizes and motion.

The global occlusion management ratio (91.4%) is very reasonable. It has been obtained after testing a total of 532 occlusions. Moreover it has been extracted in a single frame analysis, hence the results in the whole path of an object are better. The strength of the system is the use of a multi-level framework that allows to solve an occlusion from 3 different and complementary points of view.

Tracking

After detecting consecutively a cluster several times, a tracking stage, based on Kalman filter techniques, combined with a multi-frame validation process takes place. This final step is used to reinforce the coherence of the detected objects over time, obtaining a more stable position, avoiding occlusions in case the previous methods fail, and minimizing the effect of both false-positive and false-negative detections. At the same time, the tracking stage is able to manage two common problems of background subtraction: the ghosts effect and the stationary objects.

Over 2 hours of video sequences were recorded and the algorithms tested on very different situations. In general, the results depicts satisfactory detection rates and demonstrate the effectiveness of developing every module of the approach adaptive and self-regulating. The objectives proposed for the thesis have been widely achieved.

5.2 Future work

From the results and conclusions of the present work, several future lines for each treated topic are devised. They correspond to aspects that have not been solved or that need a further analysis to improve the performance of the system.

- With respect to the camera auto-calibration, an interesting improvement is related to the recalibration process in case of PTZ displacements. The idea is to develop a segment tracking, to use the same set of orthogonal lines to find the new position of the previously used vanishing points.

- About shadows, a further analysis has to be done to find a general method. Testing new ideas like probabilistic approaches are necessary to manage a problem that it has not been solved during decades.

- For the occlusion reasoning method, an online classifier with different object models could improve the results significantly.

- Due to the high diversity of camera views, operating conditions and observation objectives in traffic surveillance, there is an important lack of a common framework and most authors use their proprietary sequences. This condition has generated a large diverse body of work, where it is difficult to perform direct comparison between the proposed algorithms. It would be very important to generate a public traffic database, with a wide range of scenarios and conditions, to be able to make these comparatives.

- Talking about commercial applications, it would be a good showcase to extend the approach to night conditions or with an automatic counting system or an incident detection system.

- Finally, longer experiments under new different conditions should be performed to test the robustness of the system.

References

[1] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[2] R. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1986.

[3] Z. Kim, "Camera calibration from orthogonally projected coordinates with noisy-ransac," in *Proceedings of the IEEE Workshop on Application of Computer Vision*, 2009.

[4] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *International Journal of Computer Vision*, vol. 4, pp. 127–140, 1990.

[5] R. Cipolla, T. Drummond, and D. Robertson, "Camera calibration from vanishing points in images of architectural scenes," 1999.

[6] C. Rother, "A new approach to vanishing point detection in architectural environments," *Image and Vision Computing*, vol. 20, pp. 647–655, 2002.

[7] J. P. Tardif, "Non-iterative approach for fast and accurate vanishing point detection," in *Proceedings of the IEEE Conference on Computer Vision*, 2009.

[8] F. Lv, T. Zhao, and R. Nevatia, "Camera calibration from video of a walking human," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1513–1518, 2006.

[9] I. N. Junejo, "Using pedestrians walking on uneven terrains for camera calibration," in *Machine Vision and Applications*, vol. 22, 2009, pp. 137–144.

[10] Z. Zhang, M. Li, K. Huang, and T. Tan, "Camera auto-calibration using pedestrians and zebra-crossings," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2011, pp. 1697–1704.

[11] M. Hodlmoser, B. Micusik, and M. Kampel, "Practical camera auto-calibration based on object appearance and motion for traffic scene visual surveillance," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1–8.

[12] T. Hue, S. Lu, and J. Zhang, "Self-calibration of traffic surveillance camera using motion tracking," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, 2008.

[13] T. Schoepflin and D. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 2, pp. 90–98, 2003.

[14] N. Friedman and S. Russell, "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 1997, pp. 175–181.

[15] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 1999.

[16] D.-S. Lee, "Effective gaussian mixture learning for video background subtraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 827–832, 2005.

[17] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letter*, vol. 27, no. 7, pp. 773–780, 2006.

[18] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proceedings of the European Conference on Computer Vision*, 2000.

[19] A. Prati, I. Mikic, M. M. Tridevi, and R. Cucchiara, "Detecting moving shadows: Algorithms and evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 918–923, 2003.

[20] A. J. Joshi and N. Papanikolopoulos, "Learning to detect moving shadows in dynamic environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 2055–2063, 2008.

[21] V. Reilly, B. Solmaz, and M. Shah, "Geometric constraints for human detection in aerial imagery," in *Proceedings of IEEE European Conference on Computer Vision*, 2010.

[22] J. Jacques, C. Jung, and S. Musse, "Background subtraction and shadow detection in grayscale video sequences," in *Proceedings of IEEE Brazilian Symposium on Computer Graphics and Image Processing*, 2005.

[23] S. Atev, O. Massoud, R. Janardan, and N. Papanikolopoulos, "A collision prediction system for traffic intersections," in *Proceedings of IEEE Conference of Intelligent Robots and Systems*, 2005.

[24] T. Horprasert, D. Harwood, and L. S. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in *Proceedings of IEEE Conference of Computer Vision FRAME-RATE Workshop*, 1999.

[25] E. Salvador, A. Cavallaro, and T. Ebrahimi, "Cast shadow segmentation using invariant color features," *Computer Vision and Image Understanding*, 2004.

[26] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, and S. Sirotti, "Improving shadow suppression in moving object detection with hsv color information," in *Proceedings of IEEE Conference on Intelligent Transportation Systems*, 2001.

[27] O. Schreer, I. Feldmann, U. Goelz, and P. Kauff, “Fast and robust shadow detection in videoconference application,” in *Proceedings of IEEE International Symposium Video Processing and Multimedia Communications*, 2002.

[28] N. Martel-Brisson and A. Zaccarin, “Learning and removing cast shadows through a multidistribution approach,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1133–1146, 2007.

[29] F. Porikli and J. Thornton, “Shadow flow: A recursive method to learn moving cast shadows,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2005.

[30] A. Sanin, C. Sanderson, and B. C. Lovell, “Shadow detection: A survey and comparative evaluation of recent methods,” *Pattern Recognition*, vol. 45, no. 4, pp. 1684–1695, 2012.

[31] A. Leone and C. Distante, “Shadow detection for moving objects based on texture analysis,” *Pattern Recognition*, vol. 40, no. 4, pp. 1222–1233, 2007.

[32] B. L. A. Sanin, C. Sanderson, “Improved shadow removal for robust person tracking in surveillance scenarios,” in *Proceedings of the International Conference on Pattern Recognition*, 2010, pp. 141–144.

[33] J. Huang and C. Chen, “Moving cast shadow detection using physics-based features,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2310–2317.

[34] H. Dahlkamp, A. Ottlik, and H. Nagel, “Comparison of edge-driven algorithms for model-based motion estimation,” in *Proceedings of the International Workshop on Spatial Coherency for Visual Motion Analysis*, 2004.

[35] Z. Fan, J. Zhou, D. Gao, and Z. Li, “Contour extraction and tracking of moving vehicles for traffic monitoring,” in *Proceedings of IEEE Conference on Intelligent Transportation Systems*, 2002.

[36] C. Pang, W. Lam, and N. Yung, “A method for vehicle count in the presence of multiple-vehicle occlusions in traffic images,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 441–459, 2007.

[37] K. Otsuka and N. Mukawa, “Multiview occlusion analysis for tracking densely populated objects based on 2-d visual angles,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[38] N. K. Kanhere and S. T. Birchfield, “Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 148–160, 2008.

[39] Z. Kim, “Real time object tracking based on dynamic feature grouping with background subtraction,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[40] J. Kosecka and W. Zhang, “Efficient computation of vanishing points,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002.

[41] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[42] J.-Y. Bouguet. (2010). [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[43] R. Toldo and A. Fusiello, “Robust multiple structures estimation with j-linkage,” in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 537–547.

[44] (2013) Google maps, google. [Online]. Available: https://maps.google.es/

[45] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “Speeded-up robust features (surf),” *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008.

[46] M. Kölsch and M. Turk, “Fast 2d hand tracking with flocks of features and multi-cue integration,” in *IEEE Workshop on Real-Time Vision for Human-Computer Interaction*, 2004.

[47] E. Rosten, R. Porter, and T. Drummond, “Faster and better: A machine learning approach to corner detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 105–119, 2010.

[48] R. Montero and E. Bribiesca, “State of the art of compactness and circularity measures,” in *International Mathematical Forum*, 2009.

[49] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[50] Traffic dataset of the lunds universitet. [Online]. Available: http://www.tft.lth.se/video/co_operation/data_exchange/

[51] Image sequence server of the institut fur algorithmen und kognitive systeme, universitat of karlsruhe. [Online]. Available: http://i21www.ira.uka.de/image_sequences/

[52] Candela surveillance database. [Online]. Available: http://www.multitel.be/˜va/candela/