Post-processing of Videos for Vehicle Counting

Abel Ricardo Marcão Ribeiro

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Prof. Paulo Luís Serras Lobato Correia

Examination Committee
Chairperson: Prof. Fernando Duarte Nunes
Supervisor: Prof. Paulo Luís Serras Lobato Correia
Member of Committee: Prof. Pedro Miguel Torres Mendes Jorge

October 2014
I would like to thank Professor Paulo Correia for his guidance and patient reviewing which made possible the completion of this work.
Abstract

The analysis and monitoring of road traffic are important tools to assess the quality of the mobility of people and goods across distinct locations. This analysis and monitoring can be used to improve the flow of road traffic, by suggesting alternative routes to the driver, thus allowing a better management of time, energy and resources. This idea leads to development of Intelligent Transportation Systems, which thrive for a smarter and more coordinated road network allowing drivers to make more informed and safer decisions on their travels.

This dissertation develops a Video-based Traffic Management System which takes as input traffic surveillance video sequences available on the Internet, and uses this information to count the number of passing vehicles. The proposed implementation uses multiple Temporal-Spatial Images (TSIs) based on a video surveillance sequence. The TSIs capture vehicles’ features which can be used to count them. This traffic information can be used by traffic guidance applications to allow a better decision-making for the drivers.

The proposed system is very flexible due to using multiple virtual detections lines, allowing the selection of which road lanes are analyzed. It also has low computational requirements, allowing its usage in real-time applications. The system was developed to be able to work with low resolution videos, which are the type of videos typically provided online for the majority of the traffic surveillance cameras. The proposal includes an occlusion detection method, ensuring the system robustness, allowing to achieve high accuracies in all tested operational conditions.

Keywords

Resumo

A análise e monitorização do tráfego rodoviário são ferramentas importantes para avaliar a qualidade da mobilidade de pessoas e bens entre localizações distintas. Esta análise e monitorização podem ser usadas para melhorar o fluxo do tráfego rodoviário através da sugestão de rotas alternativas que permitam ao condutor uma melhor gestão de recursos e energia. Esta ideia leva ao desenvolvimento de Sistemas de Transportes Inteligentes, que procuram uma rede de estradas mais inteligente e coordenada que permita aos condutores tomarem decisões mais informadas e seguras nas suas viagens.

Esta dissertação desenvolve um sistema de gestão de tráfego baseado em vídeo que recorre ao processamento de vídeos de vigilância de tráfego disponíveis na Internet e utiliza esta informação para contar o número de veículos. A implementação proposta usa múltiplas imagens espaço-temporais com base nos vídeos provenientes das câmaras de vigilância. Essas imagens espaço-temporais capturam características dos veículos que são extraídas para identificar-los e contar-los. Esta informação pode ser usada em aplicações de orientação de tráfego para permitir melhores decisões por parte dos condutores.

O sistema proposto é bastante flexível devido ao uso de múltiplas linhas de detecção virtual que permitem uma seleção das faixas das estradas a analisar. O seu baixo uso de requisitos computacionais permite que seja usado em aplicações de tempo real. O sistema desenvolvido é capaz de trabalhar sobre vídeos de baixa resolução, que tipicamente são os vídeos provenientes da maioria das câmaras de vigilância de tráfego. A proposta inclui também um algoritmo de detecção de oclusão que assegura robustez ao sistema permitindo obter elevados nível de exactidão em todas as condições operacionais testadas.

Palavras-chave

# Table of Contents

Abstract ................................................................................................................................. v  
Resumo ................................................................................................................................ vii  
List of Figures ..................................................................................................................... xi  
List of Tables ...................................................................................................................... xiii  
List of Acronyms ................................................................................................................ xiv  

1 Introduction ....................................................................................................................... 1  
1.1 Overview ......................................................................................................................... 2  
1.2 Main Objectives ............................................................................................................... 2  
1.3 Contributions of the Dissertation ................................................................................... 3  
1.4 Structure .......................................................................................................................... 4  

2 State of the Art .................................................................................................................. 5  
2.1 Overview ......................................................................................................................... 6  
2.2 Automatic classification methods .................................................................................... 6  
  2.2.1 Inductive Loops ......................................................................................................... 6  
  2.2.2 Other Sensing Systems ............................................................................................ 7  
  2.2.3 Cooperative systems ............................................................................................... 8  
  2.2.4 Crowdsourcing ........................................................................................................ 9  
2.3 Image and Video methods ............................................................................................... 10  
  2.3.1 Foreground Estimation ............................................................................................ 10  
  2.3.2 Object Detection ...................................................................................................... 15  
  2.3.3 Vehicle Classification ............................................................................................. 18  
  2.3.4 Video-based Traffic Management Systems ............................................................ 19  
2.4 Operational Conditions ................................................................................................. 21  
  2.4.1 Occlusion ................................................................................................................ 21  
  2.4.2 Night time ................................................................................................................ 27  
  2.4.3 Weather ................................................................................................................... 27  
  2.4.4 Shadows .................................................................................................................. 28  

3 Proposed System .............................................................................................................. 29  
3.1 Overview ......................................................................................................................... 30
3.2 VDL Selection ........................................................................ 32
3.3 TSI Generation ..................................................................... 33
3.4 Background Estimation & Subtraction ..................................... 36
   1. TSI pixel intensity histogram .............................................. 36
   2. Column selection ............................................................ 36
   3. Column subtraction ......................................................... 37
   3.4.1 Comparison with usual column selection method .......... 38
3.5 Edge & Object detection ........................................................ 39
   1. Edge detection .............................................................. 39
   2. Object detection ........................................................... 41
3.6 Feature Extraction .............................................................. 41
   1. Generation of bounding rectangle .................................... 41
   2. Estimation of centroid .................................................... 42
   3. Acquisition of objects’ features ....................................... 42
3.7 False Vehicle Elimination ..................................................... 42
   1. Verification of objects’ edges .......................................... 43
   2. Verification of objects’ area .......................................... 44
   3. Verification of overlapping rectangles ......................... 45
3.8 Occlusion detection ........................................................... 46
   1. Generation of a second VDL .......................................... 48
   2. Determination of first VDL crossed ............................... 49
   3. Search for possible occlusions ..................................... 50
3.9 Vehicle Counting .............................................................. 51
   1. Selection of one TSI as reference .................................... 52
   2. Initial vehicle count ...................................................... 52
   3. Relaxed search to link possible occlusions ................... 52
3.10 Cumulative data treatment .................................................. 54
3.11 Detection Optimization ...................................................... 55
   3.11.1 Mode Selection ...................................................... 56
   3.11.2 Settings of the modes ............................................ 57
   3.11.3 Sensitivity threshold optimization ........................... 60
4 Experimental Results .......................................................... 63
4.1 Overview ............................................................................. 64
4.2 Test Conditions ............................................................... 64
   4.2.1 Videos with good illumination .................................. 64
   4.2.2 Videos with jitter .................................................... 65
   4.2.3 Videos with shadowed areas ................................. 65
4.2.4 Videos with rain ................................................................. 66
4.2.5 Blurred videos................................................................................ 66
4.2.6 Videos with moving camera ......................................................... 67
4.2.7 Dusk/Dawn video ...................................................................... 67
4.2.8 Night videos ................................................................................. 68

4.3 Vehicle Count Evaluation .......................................................... 68
4.3.1 Day mode (DM) ........................................................................... 69
4.3.2 Night modes (NBSM & NFSM) .................................................... 72
4.3.3 Cumulative data treatment results ............................................. 73

4.4 Computational Performance ....................................................... 74
4.5 Results comparison ................................................................. 75
4.6 Performance with 3 or more VDLs ............................................ 75

5 Conclusions .................................................................................. 77
5.1 Overview .................................................................................... 78
5.2 Strengths .................................................................................. 78
5.2.1 Flexibility .................................................................................. 78
5.2.2 Easy -to-use ............................................................................... 78
5.2.3 Performance & Efficiency ......................................................... 78
5.3 Weaknesses ................................................................................ 79
5.3.1 Camera Movements ................................................................. 79
5.3.2 Camera Position & Angle ............................................................ 79
5.3.3 Light reflections & shadows ...................................................... 79
5.4 Future Work ............................................................................... 80

Annex 1-Graphical Interface .......................................................... 81
Annex 2 - Videos List ......................................................................... 86

References .......................................................................................... 92
List of Figures

Fig.1: Example of inductive loops [U.S. Gov, 2014] .................................................7
Fig.2: Traffic information provided by the INRIX crowdsourcing system [INRIX, 2014] .................9
Fig.3: Information provided to the driver by the Waze crowdsourcing system [Waze, 2014] ...........9
Fig.4: Block diagram of a traffic management system ..................................................10
Fig.5: Estimation of the foreground mask ....................................................................11
Fig.6: Diagram of Object Detection ............................................................................15
Fig.7: Relation between polar, \((r, \theta)\), and Cartesian, \(x, y\), coordinate spaces ......................17
Fig.8: Diagram of Vehicle Classification ......................................................................19
Fig.9: Example of toll station with ANPR [Brisa, 2014] ...............................................20
Fig.10: Left: VDL in sample image; Centre: Resultant TSI; Right: Resultant TOB .................20
Fig.11: Thermal image provided by a thermal camera [FLIR, 2014] .................................21
Fig.12: Example of occlusion in a LTSV camera ............................................................22
Fig.13: Left: result of SPCPE; centre: result of NSS; right: result of size-adjustment [Liu, 2011] .......24
Fig.14: Section of scaled version of multiple TSIs generated from MVDLs [Mithun, 2012] ..........26
Fig.15: Difference between night time and day time images, with clean weather .................27
Fig.16: Difference between rainy and clean weather images ............................................27
Fig.17: Left: VDL on sample image; Right: Resultant TSI with object detection showing occlusion ...28
Fig.18: Diagram with a summary of each step of the system ...........................................31
Fig.19: Bresenham’s line algorithm example [Crotalus, 2014] ............................................32
Fig.20: Interface to define the VDLs by the user ............................................................33
Fig.21: Flowchart of the TSI Generation .......................................................................34
Fig.22: Pixel assortment in the vector \(v_{jk}\) ...................................................................35
Fig.23: Left: VDL in video’s sample; Right: Corresponding TSI according to its length and video’s time. ........................................................................35
Fig.24: Flowchart of background estimation and subtraction .............................................36
Fig.25: Left: VDL used for illustration of background subtraction operation; Center: proposed method; Right: minimum variance method ..................................................38
Fig.26: Flowchart of Edge and Object Detection ............................................................39
Fig.27: Flowchart of Feature Extraction ........................................................................41
Fig.28: Left: Contours detected; Right: TSI with bounding rectangles with highlighted centroids. .....41
Fig.29: Flowchart of FVE ............................................................................................43
Fig.30: Left: TSI without edges verification; right: TSI with edges verification .......................44
Fig.31: Left: \(\alpha = 0.25\); right: \(\alpha = 0.5\) ................................................................45
Fig.32: Left: Vehicle detection result without and with FVE, respectively on left and right ..........46
Fig.33: Right: \(\beta = 0.65\); Left: \(\beta = 0.35\) ..................................................................46
Fig.34: Flowchart of object occlusion detection ..................................................................48
Fig.35: Examples of second generated VDL position .........................................................49
Fig.36: Example of object association .............................................................................50
Fig.37: Left: Position of the lines in the video; Center: Green line contour detection and TSI with occlusion; Right: Red line contour detection and TSI with the detected occlusion .................................................51
Fig.38: Flow chart of vehicle count acquisition ................................................................. 52
Fig.39: Flow chart of state detection .................................................................................. 56
Fig.40: Double detection due to headlights’ reflection ......................................................... 56
Fig.41: Canny threshold curve on day mode ..................................................................... 58
Fig.42: Canny threshold curve on night modes ................................................................. 58
Fig.43: Examples of video with good illumination during the day and night ..................... 65
Fig.44: Examples of jitter in videos ................................................................................... 65
Fig.45: Examples of videos with shadowed area ............................................................... 66
Fig.46: Examples of rainy videos ....................................................................................... 66
Fig.47: Examples of blurred videos ................................................................................... 67
Fig.48: Example of video camera shaking ......................................................................... 67
Fig.49: Examples of dusk videos ....................................................................................... 68
Fig.50: Examples of night videos ....................................................................................... 68
Fig.51: Examples of Day mode results ............................................................................... 70
Fig.52: Examples of TSIs in rainy and blurry videos .......................................................... 71
Fig.53: Examples of markings detection in TSIs of shaking videos .................................... 71
Fig.54: Examples of duck detection issues ......................................................................... 71
Fig.55: Example of detected occlusion ............................................................................. 72
Fig.56: Example of undetected occlusion ....................................................................... 72
Fig.57: Examples of NFSM double detections .................................................................. 73
Fig.58: Example of failed search for vehicle on TSIs ......................................................... 76
Fig.59: Examples of cameras with different heights and angles ......................................... 79
Fig.60: Examples of light reflections and shadows .............................................................. 80
Fig.61: Main window ......................................................................................................... 84
Fig.62: Canny settings window ......................................................................................... 85
Fig.63: Parameters window ............................................................................................. 85
List of Tables

Table 1: Day mode accuracies........................................................................................................70
Table 2: NBSM and NFSM accuracies..............................................................................................72
Table 3: Accuracy with trained and untrained data ...........................................................................74
Table 5: Results comparison .............................................................................................................75
Table 6: Accuracies for 2,3 and 4 VDLs ..........................................................................................76
Table 7: List of DM videos ................................................................................................................88
Table 8: List of NBSM videos ............................................................................................................90
Table 9: List of NFSM videos ............................................................................................................91
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DM</td>
<td>Day Mode</td>
</tr>
<tr>
<td>EM</td>
<td>Expectation-Maximization</td>
</tr>
<tr>
<td>FVE</td>
<td>False Vehicle Elimination</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HSV</td>
<td>Hue, Saturation and Value</td>
</tr>
<tr>
<td>LTSV</td>
<td>Live Traffic Surveillance Video</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>MVDL</td>
<td>Multiple Virtual Detection Line</td>
</tr>
<tr>
<td>NBSM</td>
<td>Night Back Side Mode</td>
</tr>
<tr>
<td>NFSM</td>
<td>Night Front Side Mode</td>
</tr>
<tr>
<td>NSS</td>
<td>N-Steps Search</td>
</tr>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>SPCPE</td>
<td>Simultaneous Partition and Class Parameter Estimation</td>
</tr>
<tr>
<td>TOB</td>
<td>Temporal-Spatial Image Object Blob</td>
</tr>
<tr>
<td>TSI</td>
<td>Temporal-Spatial Image</td>
</tr>
<tr>
<td>VDL</td>
<td>Virtual Detection Line</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This first chapter gives a brief overview of the work. It establishes the work main objective, original contributions, the scope and motivations. The work structure for the rest of this dissertation is also provided.
1.1 Overview

The increasing number of circulating vehicles has called for the development of more efficient systems to classify and monitor road traffic. The information acquired by these methods is used to plan and to manage the road network, to allow monitoring of the vehicles on the network and to provide real-time data to guide the users of the network. These systems enhance the capacity of the road network infrastructure improving drivers’ safety and traffic management efficiency.

Traffic management with real-time data gains special importance in densely populated areas, due to the high amount of vehicles circulating in them, as it provides the means to better assess traffic conditions and thus be able to divert it in case of unpredictable events such as car accidents, which may cause long lines of vehicles and traffic jams, eventually leading to more accidents due to the confusion originating from these events. Hence, the importance of Intelligent Transport Systems (ITSs) development, which should be fast enough to reach and divert the traffic in such events [EU M&T, 2014].

ITSs are defined as advanced applications which aim to provide services related to distinct modes of transportation and traffic management to allow the users to be better informed and make better use of the transport network [EU ITS, 2014]. ITSs use many different technologies, which include traffic signal control systems, traffic management systems or video-based systems from Live Traffic Surveillance Video (LTSV) cameras.

The decrease in hardware costs caused an increase in the number of video cameras used to monitor the existing infrastructures, providing a subsequent improvement of the accessibility to data, allowing the fast development of computer vision methods. These methods can be used in a variety of applications, which include congestion classification, traffic rule violation and vehicle interaction, thus having an increasing relevance for ITSs. These applications can be deployed in systems that were traditionally developed to be used by a human operator, lowering the cost for its implementation. These automatic video-based systems provide the human operator with traffic information, such as vehicle counting, from the different cameras which the operator has to analyze, thus avoiding possible errors made by the traditional manual counting done by this operator. Hence, these automatic systems which process the videos provided by LTSV cameras for traffic assessment are the main focus of this dissertation.

Section 1.2 details the main objectives of the dissertation. It is followed by section 1.3 which enumerates the main contributions and finalizes with an outline of the work in section 1.4.

1.2 Main Objectives

The use of video cameras for monitoring purposes is becoming more popular overtime. Hence, in this dissertation a video-based road traffic monitoring system is developed. This system provides a traffic assessment through vehicle counting using video and image processing methods. Any device such as a GPS or a cell phone could receive the result of the processed traffic information and use it
to decide the best route to take and thus avoid traffic congested paths.

The system should be robust, so that it can automatically adapt to different operational conditions on the same scene, such as weather or illumination changes along the day, without human intervention, although a human operator should also be able to adjust its parameters manually when he thinks it is fit. The interface which the human operator has access should allow him to easily choose which lanes of the road should be analyzed, receive the information from those locations in a simple manner and have easy access to the parameters being used by the system at any time.

The parameters, which are adapted on distinct conditions, are mainly detection parameters used by the employed filters and the edge detector, parameters used in the false vehicle detection module to validate the objects detected as vehicles, and parameters to find possible occluded vehicles.

This dissertation intends to develop a prototype software where this road traffic monitoring system is implemented. The software should be user-friendly to allow easy observation of the results and the analysis done to the source video from the LTSV, in order to allow adjustments when the operator deems necessary.

1.3 Contributions of the Dissertation

The main contributions of this dissertation can be enumerated as follows:

1. The development of a system which allows the monitoring of road traffic through the use of an automatic video-based system, using LTSV cameras as the source of information. This information is extracted from Temporal-Spatial Images generated from the video data provided by the LTSV cameras.
2. The system implemented is robust, adapting itself to different operational conditions on the scene being captured, such as night and day time and inclement weather conditions, without any human operator interference.
3. The methods used allow the data provided from cameras to have poor quality as the system is implemented to work with low resolution traffic cameras.
4. The traffic monitoring system has the flexibility to allow the human operator to assess the traffic information in any individual road lanes of his choosing or in all the lanes at the same time, which allows the verification of possible problems in some lanes, such as possible bottlenecks.
5. The friendly GUI interface allows the visualization of the results of distinct phases of the vehicle counting process and also the visualization of the final results. This helps to understand the whole vehicle counting process.
6. The system is developed to minimize the possibility of missing occluded vehicles and the detection of false vehicles through the use of a novel occlusion detection method.
7. The development of a detection system using three modes which allows improved vehicle detections in scenes with different levels of illuminations.
8. The use of a distinct background estimation approach which improves the background subtraction.
9. The development of methods of false vehicle detection to reject object detections which do not represent vehicles.
10. The system uses a cumulative data approach which improves the accuracy of the vehicle counting by using the data provided by a high number of videos to know the approximate features of the vehicles.

1.4 Structure

This dissertation is composed of 5 Chapters, where the 1st Chapter is this Introduction of the work.

The 2nd Chapter details the State of the Art related with traffic management systems, explaining the non-video based methods in section 2.2, video-based methods in section 2.3, also detailing the whole analysis process and giving examples of existing systems, and finalizes with a discussion on the operational conditions to which traffic management cameras are subjected on a daily basis.

The 3rd Chapter is dedicated to presenting and explaining the various functionalities of the traffic monitoring system. The first nine sections introduce and detail in sequential order each distinct process of the system until the final vehicle count is achieved. The last two sections, namely 3.10 and 3.11, respectively describe an optional approach using known information relative to the objects’ features to improve the accuracy of vehicle counting and an optimization process which can be used to improve the detection of the vehicles in the presented system.

The 4th Chapter is devoted to the experimental results obtained with the developed system. The chapter starts by explaining the parameters chosen by default on the system and by defining the operational conditions of the videos used on the experiments. The evaluation of the vehicle count results are discussed in section 4.3 and the computational performance of the system are evaluated in section 4.4.

The 5th Chapter presents the conclusions of this work, pointing its strengths and weaknesses and detailing some possible future work.
This chapter provides an overview of the state of the art related with traffic management systems.
2.1 Overview

Traffic management is a field primarily dedicated to improve the flow of vehicles and improve road's safety. It involves traffic monitoring and classification by local or national roadway authorities to manage traffic flows and provide advice concerning traffic congestion. The traffic flow classification can be divided into manual, semi-automatic and automatic.

Manual classification requires the intervention of human operators, such as the observation of roads by traffic reporters or traffic patrols. These methods employ a minimal amount of technology and rely mainly on the operator. The automatic and semi-automatic classifications try to replicate the human decisions taken by manual classifications.

Semi-automatic classification requires the intervention of the end-user for its efficient operation. This classification uses information such as the speed limit and the average speed of the users on the roads to determine the traffic congestion along the day. However, information such as average speed has to be voluntarily provided by a considerable amount of users for an accurate estimate the traffic on distinct weekdays.

Automatic classification doesn’t require any assistance from a human operator for the computation of traffic information. The input information needed for this classification can be obtained through the use of devices such as inductive loops or sensors, as discussed in section 2.2. Also image and video analysis methods can be used for the automatic traffic flow classification, being discussed in detail in section 2.3, as they are the main focus of the dissertation. The operational conditions which video-based systems have to deal with are detailed in section 2.4.

2.2 Automatic classification methods

Automatic traffic flow classification methods are slowly replacing the traditional manual methods, mostly due to the widespread adoption of new monitoring and tracking technologies over the years. These technologies vary in cost and accuracy and are discussed in the following sub-sections, excluding those relying on image and video analysis, which are described in section 2.3.

2.2.1 Inductive Loops

An inductive loop is an electromagnetic system which uses a moving magnet to induce an electrical current in a nearby wire. It perceives the presence of a conductive metal object by inducing currents in the object, which change the loop inductance. Inductive loops for traffic management applications are made of insulated, electrically conducting wire, typically installed under the road pavement. These loops are usually placed in motorways or near an approaching traffic light. When a vehicle passes over the loop it causes a reduction in the loop inductance due to the vehicle eddy currents. This reduction will make the loop electronic output send a pulse to the traffic signal controller signaling the presence or passage of a vehicle. An example of an inductive loop installed on the road
pavement is shown in Fig. 1, to count the vehicles that cross each lane. This is the most commonly used method by road exploration entities to count vehicles due to its high accuracy, however it is a very intrusive method and it has a high maintenance cost. [U.S. Gov, DT]

![Fig.1: Example of inductive loops [U.S. Gov, 2014]](image)

2.2.2 Other Sensing Systems

The advances in electronics research have provided cheaper and better performing sensors over the years. Sensors of many types are already being extensively used in vehicles to improve the navigation and thus the driver’s safety.

Floating probe data collection

Floating probe data collection is a set of low-cost sensory methods that allow the estimation of speed and travel time data along any type of transportation routes. The raw data can be obtained mainly by three distinct methods, which are subsequently explained: the triangulation, vehicle re-identification and GPS-based methods.

Triangulation method: There are a high proportion of vehicles that usually contain at least one cell phone, especially in developed countries. These cell phones transmit at certain intervals their presence information to a network of antennas. When a vehicle moves, so does the signal of any cell phones that are inside the vehicle. Using a triangulation method with the received data from the antennas, it is possible to know the approximate position of the vehicle and its direction and speed. The status of flowing traffic can be approximately estimated by counting the number of cell phones on a given road. The main advantage of this method is that no new infrastructure is needed along the road, as it only uses the cell phone network antennas for the triangulation. However, it not a very precise method as there is no guaranty that each vehicle just has one cell phone, for instance, a bus will probably have many cell phones inside thus it will be considered by this method as many vehicles
instead of just one. Moreover, the pedestrians walking near the road cause further errors on the traffic assessment. The triangulated position can also be difficult to obtain due to phone towers that serve large areas, being prone to position errors.

**Vehicle re-identification:** Vehicle re-identification methods use sets of electronic beacons or lasers along the road. These methods have a unique serial number of one device inside the vehicle and detect it in one location and later detect it once more, re-identifying it, on the same road. This communication between vehicle-to-infrastructure and infrastructure-to-vehicle allows the computation of the speed and travel time of the vehicle along the road. Therefore, it basically uses pairs of sensors to identify vehicles. The identification can be a Media Access Control (MAC) address from Bluetooth devices or serial numbers from electronic toll collection transponders. These systems have a good accuracy if the vehicle does not cross the sensors at an extremely high speed. However, it is necessary to equip the vehicles with a unique identifier for this purpose.

**GPS-based methods:** The GPS-based methods use a device installed in the vehicle that communicates with geostationary satellites to determine its position and speed. It needs to be connected to at least four satellites to allow the determination of the position \((x, y, z)\) and the time delay between the satellites internal clocks and the GPS device clock. The use of the position information computed by the GPS devices placed in vehicles can be sent to a data center and used to analyze the traffic. The data should be processed according to the day of the week and time of the day, to allow the assessment of the traffic at different hours on distinct days. This processed data provides information about which roads should be avoided at distinct times of the day, and thus can be used to avoid sending GPS users through usually congested routes. The method has a worst case accuracy of 7.8 meters at a 95% confidence level [GPS Acc, 2014], therefore it allows for a good determination of the position of the vehicle. However, the device needs to be in communication with 4 satellites to allow the computation of the data. Furthermore, in cities with tall building this communication may have obstructions and a data control center is necessary to integrate all the data provided by the GPS devices.

### 2.2.3 Cooperative systems

Cooperative systems pursue to improve the on-board assistance of the driver through communication vehicle-to-vehicle and vehicle-to-road-infrastructure which can help drivers to improve the control of their vehicle and therefore have positive effects in terms of safety and traffic efficiency. These systems are based on the transfer of real-time information to the vehicles via radio interfaces. The information provided to the drivers comes from data centers, which ideally receive data from distinct sources, such as weather forecast stations, traffic departments or position information from the vehicles, where it is integrated and processed. Thus, these systems can provide information about route recommendations, alerts and warnings due to traffic jams, road works or accidents. It is still a relatively new type of system and its development is being supported by the European Union in large scale pilot projects such as EuroFOT and Drive C2X [EU ITS, 2014].
2.2.4 Crowdsourcing

Crowdsourcing is based on the idea of obtaining a needed service or content by soliciting information from a large group of people. This crowd contributes voluntarily information to a data center which processes it and offers the processed results back to its contributors. In the case of crowdsourcing related to traffic management, the users have a variety of mobile applications and internet services which allow them to provide traffic feedback to specialized companies, such as INRIX [INRIX, 2014], which is illustrated in Fig.2 with the map of London traffic and Waze [Waze, 2014]. As a result of being contributors, users are provided with real traffic information, travel times and traffic forecasts. The drawback of this system is that it is dependent of the fidelity of the information offered by the contributors, which may not be completely correct.

Fig.2: Traffic information provided by the INRIX crowdsourcing system [INRIX, 2014]

Fig.3: Information provided to the driver by the Waze crowdsourcing system [Waze, 2014]
2.3 Image and Video methods

In recent times, the use of automatic video analysis using Live Traffic Surveillance Video (LTSV) cameras has emerged using image and video methods for traffic management. These cameras are positioned in poles that are higher than standard Closed-circuit television (CCTV) cameras and are harder to install. The extra height allows a better viewing angle, which reduces the occlusion between densely spaced vehicles [Buch, 2011]. The increasing number of LTSV cameras and improved infrastructures created for this purpose increased the amount of data available for video analysis. Moreover, the development of analytical techniques to process videos and the increasing computer power has allowed more robust applications to be created. These video analysis methods aid the human operators providing useful information automatically extracted from the cameras surveying the road traffic.

A generic architecture of such methods is illustrated in Fig.4. These methods usually start by a foreground estimation of video or image, by removing the background scene and thus highlight the moving objects that are considered candidate vehicles. In this phase a background model is typically used. The next step receives the mask of the foreground estimation, to detect the moving objects. A detection algorithm is used to identify the objects, generating an object list, and extract the relevant features from them. In a third phase, the objects in this list are classified has vehicles or not, based on a set of defined characteristics that should distinguish the vehicles from any other possible moving object thus avoiding false positives. As an output of this phase, we should have a list of vehicle allowing their counting and, depending on the algorithms used, their tracking.

Fig.4: Block diagram of a traffic management system

The next three sections, 2.3.1, 2.3.2 and 2.3.3, start by describing each of these phases in detail and overview some commonly used methods. Subsequently, examples of existing traffic management systems are provided in section 2.3.4.

2.3.1 Foreground Estimation

In image and video processing, object motion detection is a process that finds moving instances of real-world objects, such as human beings or animals, in images or video streams. The motion detection methods usually try to highlight the moving objects from the rest of the scene of the
source. Afterwards, features are often extracted from these objects to help categorize them with the help of learning algorithms.

There is an extensive amount of research in object motion detection due to the high amount of applications it has, also related with the automotive industry, especially to increase road safety. Since in this dissertation we are working to detect and count vehicles using LTSVs cameras, the methods addressed will be related with the detection of objects in motion, using what is usually called Background or Foreground Estimation.

The methods for this estimation vary mainly on the model used to generate the background model, as there are several approaches to perform this model. The idea behind them is detecting the moving objects from the difference between a reference frame, usually called background image, and the current frame. Hence, the background image is subtracted from the current frame highlighting the moving objects. The background image must be as close as possible to the representation of the scene of the video with no moving objects and must be regularly updated so it can adapt to possible luminance changes or changes of the background objects. The idea is illustrated in Fig.5.

![Fig.5: Estimation of the foreground mask](image)

The following sub-sections discuss three of the most commonly used approaches, followed by a discussion comparing their performance and a small summary of other existing foreground estimation methods.

2.3.1.1 Running Gaussian average

The Gaussian average method is the most popular method for detection of moving objects due to its simplicity. This method, developed by Wren et al. [Wren, 1997], uses a model to generate an independent background at each pixel location, \( I(i,j) \). This model uses a Gaussian Probability Density Function (PDF) based on the last \( n \) values at each pixel location. The model is updated with each new frame analyzed. To that effect it keeps an on-line cumulative (running) average, to avoid fitting the pdf from scratch at every new frame, following the equation 2.1:

\[
\mu_t = \alpha I_t(i,j) + (1 - \alpha)\mu_{t-1} \tag{2.1}
\]
where:

- \( I_t(i,j) \) is the current value of the pixel in position \((i,j)\);
- \( \mu_{t-1} \) is the average at time \( t-1 \);
- \( \alpha \) is a value in \([0,1]\).

The value \( \alpha \) is chosen as a tradeoff between quick update and stability. The higher the value, the faster is the update of the background. A pixel \( I_t(i,j) \) will be considered foreground at frame \( t \) if the satisfies the inequality 2.2:

\[
|I_t(i,j) - \mu_t| > k\sigma_t
\]

where:

- \( k \) is an arbitrary value;
- \( \sigma_t \) is the standard deviation of the Gaussian PDF. The estimate of \( \sigma_t \) can be obtained much in the same way as \( \mu_t \).

However, the model can suffer from an excess of updates in the presence of foreground objects as remarked in [Koller, 1994]. Therefore, they propose a modification on the model update as follows:

\[
\mu_t = M\mu_{t-1} + (1-M)(\alpha I_t(i,j) + (1-\alpha)\mu_{t-1})
\]

Where:

\[
\begin{cases} 
M = 1, & \text{if it is a foreground value} \\
M = 0, & \text{if it is a background value}
\end{cases}
\]

The real-time requirement constrains of this model involve mainly the computation of \( \mu \) and \( \sigma \) and the classification of pixels as foreground or not, thus the computational load will increase with higher frequencies of updates of these values. Usually the background subtraction is applied to the luminance of a grayscale image; nonetheless this model can also be used for multiple-component color spaces, such as \( HSV \) or \( RGB \), increasing the computational load due the higher number of dimensions.

### 2.3.1.2 Mixture of Gaussians

The running Gaussian average adapts correctly to slow background variations. However, if the changes in the background are fast and periodic, such as the movement of the leaves of a tree due to the wind or the ripple-wave pattern seen in the water due to the waves, these variations may be incorrectly considered as moving objects. The model presented in [Stauffer, 1999], is able to cope with this issue using multiple background models with distinct probabilities.

The probability model to describe the chances of observing a certain pixel of value \( x \) at time \( t \) using a mixture of Gaussians as follows:
Each of the $K$ values corresponds to a distribution describing one of the possible observable backgrounds, usually with $K$ being set to a value between 3 and 5. In case of using multiple channels, such as RGB is used, the Gaussians will be multiple-variate do describe it. $\Sigma_{it}$ is the co-variance matrix of the distribution; it will be a simple diagonal if the samples are assumed independent. The peak amplitude, $\omega_i$, is used with the standard deviation, $\sigma_i$, as a criterion to discriminate the background from the foreground. The assumption is that the higher and more compact the distribution, the more likely it is to be part of the background. Hence, the first $L$ distributions that satisfy inequation 2.5 are accepted as background, where $T$ is an arbitrary chosen threshold.

$$\sum_{i=1}^{L} \frac{\omega_i}{\sigma_i} > T$$  \hspace{1cm} 2.5

With each new frame $t$ two problems have to be solved simultaneously. Each pixel observed must be matched with the best available distribution and the updated model parameters have to be estimated. [Stauffer, 1999] proposes solving these concurrent problems using an approximated version of an Expectation-Maximization (EM) algorithm with a buffer of the $n$ last frames. It uses the first in ranking order to satisfy 2.6 as the accepted match for pixel intensity $x$ at interval $t$.

$$\frac{x_t - \mu_{it}}{\sigma_{it}} > 2.5$$  \hspace{1cm} 2.6

### 2.3.1.3 Kernel Density Estimation

The background PDF can be approximated by a histogram of the most recent values classified as background values, instead of the use of a Gaussian PDF. However, due to a limited number of samples, this approximation may provide a poor modeling of the true unknown PDF. In order to tackle this issue, Elgammal et al. [Elgammal, 2000] described a model of the background subtraction using a non-parametric Kernel Density Estimation (KDE) on the buffer of the last $n$ background values. This KDE generates a smoothed and continuous version of the histogram.

The background PDF is given by the sum of Gaussian kernels centered in the most recent $n$ background values of the pixel $x_i$:

$$P(x_i) = \frac{1}{n} \sum_{i=1}^{n} \eta(x_i - x_i, \Sigma_i)$$  \hspace{1cm} 2.7

This model may seem to be just dealing with a sum of Gaussians as in section 2.3.1.2,
however it has substantial differences. In 2.3.1.2, each Gaussian describes the pdf of the pixels over time and it is updated with each new frame. In here, the co-variance $\Sigma_t$, which is usually considered diagonal as a simplification, is the same for all kernels and each Gaussian describes just one sample of data, which should have a $n$ in the order of the 100. The estimation of $\Sigma_t$, which is a key problem in KDE, is acquired by analyzing the set of differences between two consecutive values in the time domain.

The classification of a certain pixel $x_t$ as background or foreground can be simply stated as:

$$P(x_t) < T$$  \hspace{1cm} 2.8

where $T$ is an arbitrary threshold value.

The model update is simply obtained updating a buffer with the histogram of the last background values by first in first out order applying a selective update as described at the end of section 2.3.1.1 to prevent the inclusion of foreground pixels in the buffer.

2.3.1.4 Performance comparison

The complexity of the methods discussed can be compared in terms of their speed and memory requirements. Among these methods, Background Subtraction has the lowest complexity. For each pixel the classification is made using just a simple threshold difference and the background just has to adapt to the $\mu$ and $\sigma$ parameters of the model. The mixture of Gaussians has a complexity $K$ times superior to the Background Subtraction, where this $K$ is the number of Gaussian distributions used, which is usually between 3 and 5. The KDE model computes its value in the Gaussian kernels centered on $n$ frame, hence it was a complexity approximately $n$ times higher than the Background Subtraction, where $n$ is usually around 100.

2.3.1.5 Other methods

Kalman Filter: A Kalman Filter can be used to estimate a background image, where each pixel is modeled by a temporary filter. The foreground is interpreted as noise for the filter state. However, illumination changes are non-Gaussian and may cause errors, because they violate the basic assumptions for the use of Kalman filters [Messelodi, 2005].

Optical flow estimation: The proposal of [Yalcin, 2005] presents a robust method that works even when the camera moves periodically. It uses a Bayesian framework for estimating dense optical flow over time that explicitly estimates a persistent model of background appearance. However, it has a high computational load.

Eigenbackgrounds: The approach presented in [Oliver, 2000] is based on an eigenvalue decomposition applied to the whole image and not only blocks. This extended spatial domain can explore possible spatial correlation, updating it over time, and avoid the tiling effect of block partitioning.
Co-occurrence of image variations: The method presented in [Seki, 2003] focuses on the idea of exploiting spatial interdependency of the image. The main assumption is that neighboring blocks should experience similar variations over time. Although this assumption holds true for blocks belonging to the same background object, it obviously is not true for blocks on the edge of distinct objects. Therefore, it is prone to false object detections near those edges.

2.3.2 Object Detection

In the object detection module, the mask computed by the foreground estimation module is received and processed in order to individualize the objects it contains. These objects are the regions of interest contained in the mask. The detection methods find the edges of those regions and generate a contour around each of them, which have to subsequently be uniquely identified. The process is illustrated in Fig.6. There are several distinct methods which allow this detection, a selection of which are discussed in the following sub-sections.

![Diagram of Object Detection](image)

Fig.6: Diagram of Object Detection

2.3.2.1 Harris Edge Detector

The Harris Edge Detector searches for the intersections of two edges in the regions of interest of the mask, and thus finds the corners of the regions; hence this method is also called Harris Corner Detector.

The corners represent variations in the gradient of the image, thus if we consider a grayscale image \(I\) and apply a window \(w(i,j)\), we can compute the variation of the intensity as:

\[
E(u,v) = \sum_{ij} w(i,j)[I(i + u, j + v) - I(i,j)]^2
\]

where:

- \(u\) and \(v\) are displacements, respectively, in the \(i\) and \(j\) directions;
- \(E(u,v)\) is the intensity variation with displacement \(u\) and \(v\);
- \(I(i,j)\) is the intensity at \((i,j)\);
- \(w(i,j)\) is the window at position \((i,j)\);
- \(I(i + u, j + v)\) is the intensity at a moved window \((i + u, j + v)\).

The quadratic power placed in the expression \(I(i + u, j + v) - I(i,j)\) accentuates the large
variations in the intensity of the windows.

Using Taylor expansion and then expressing the result in a matrix form we have:

\[
E(u, v) = [u \; v] \left( \sum_{i,j} w(i,j) \begin{pmatrix} I_i^2 & I_i I_j \\ I_j I_i & I_j^2 \end{pmatrix} \right) [u \; v]
\]

where \(I_i\) and \(I_j\) are the derivatives of \(i\) and \(j\) in respect to the image \(I\).

Considering matrix \(M\):

\[
M = \sum_{i,j} w(i,j) \begin{pmatrix} I_i^2 & I_i I_j \\ I_j I_i & I_j^2 \end{pmatrix}
\]

We can compute the score for each window:

\[
R = \text{det}(M) - k(\text{trace}(M))^2
\]

Finally, this score \(R\) is compared with an arbitrarily chosen value \(T\).

\[
\begin{cases} 
  \text{if } R < T, & \text{the pixel } I(i,j) \text{ is considered a corner} \\
  \text{if } R > T, & \text{the pixel } I(i,j) \text{ is not considered a corner}
\end{cases}
\]

Thus the adjustable parameter in this method is just the threshold \(T\). In traffic management systems, this parameter may have to vary according to the operational condition, since they vary the intensity of the pixels in the image.

### 2.3.2.2 Canny Edge Detector

The Canny Edge Detector is a multi-step method which detects the edges of the regions of interest using gradient variations in the mask. This detector is known as the optimal detector and it tries to satisfy three criteria:

- Low error rate: to avoid detection non-existent edges;
- Minimal response: to have just one response for each edge;
- Good localization: the distance between the real edge and the pixel edge detected should be the smallest possible [OpenCV, Doc].

This method takes four steps, starting with a filter to remove some of the noise from the image. It is followed by a second step where the intensity gradient of the image is found using two convolution masks \(G_x\) and \(G_y\) to detect the intensity variation along the two axis of the image and computing its strength and direction as shown below, rounding the directions to 0, 45, 90 or 135. The third step applies a non-maximum suppression to remove pixels that are not considered to be part of an edge,
hence only remaining the candidate edge lines.

Gradient strength and direction:

\[ G = \sqrt{G_x^2 + G_y^2}, \quad \theta = \arctan \left( \frac{G_y}{G_x} \right) \]  \hspace{1cm} (2.14)

The final step uses two thresholds to decide which lines should be considered edges. If a pixel gradient is higher than the upper threshold, the pixel is accepted as an edge. If a pixel gradient value is below the lower threshold, then it is rejected. If the pixel gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the upper threshold.

In traffic management systems, the Canny parameters, which can be adjusted according to the operational conditions are obviously the lower and upper threshold, but also Canny’s aperture, which changes the sensitivity of the detector to gradient changes. The filter and its size can also be adapted according to the situation.

2.3.2.3 Hough Line Transform

The Hough Line Transform is used to detect straight lines from the regions of interest of an image. The Hough transform uses a Polar coordinate system instead of the Cartesian coordinate system. Hence, for a general \((x, y)\) it is possible to define the lines that go through that point as:

\[ r_{\theta} = x \cdot \cos(\theta) + y \cdot \sin(\theta) \]  \hspace{1cm} (2.15)

Which is illustrated in Fig.7.

![Fig.7: Relation between polar, \((r, \theta)\), and Cartesian, \((x, y)\), coordinate spaces](image)

This method finds all the lines that cross a point \((x, y)\) and does this operation for all the points.
in the image. The resulting lines, following the equation for $r_{\theta}$, will be sinusoids. If these sinusoids intersect in the plan $\theta - r$ then those points belong to the same line. Therefore, it means that a line can be found by knowing the number of intersections between curves. The more points the line has in the image, higher will be the number of sinusoids intersecting each other found. Hence, this method uses a minimum threshold of number of intersections to consider a line in the image.

The Hough Line Transform has to keep count of the number of intersections between sinusoids in every point of the image and then use the threshold to find which points have a number of intersections above the threshold and consider them as lines thus generating the contours around the objects. Therefore, in traffic management systems, the adaptable parameter for different operational condition will be this threshold.

2.3.2.4 Other methods

**Generalized Hough Transform:** This method, developed in [Ballard, 1981], follows an idea quite similar to the Hough Line Transform, however it uses a voting procedure done in a parameter space from which objects with the desired shapes are obtained as local maxima of the space.

**Marr-Hildreth Operator:** The edge detection, proposed in [Marr, 1980], is based on zero-crossings of the Laplacian operator applied to a Gaussian-smoothed image. However, this operator tends to return false edges due to local minima and also gives a poor localization at curves edges, hence it is not commonly used.

2.3.3 Vehicle Classification

The Vehicle Classification step, in generic object recognition, is usually composed by a learning algorithm, which due to training with previous samples from vehicles can classify the new objects based on features previously defined and extracted from the object.

A classification module can arrange the objects into categories such as brand or type of vehicle. It can also discard objects that are not considered vehicles, known as false vehicle detection. However, there is a big difference between traffic monitoring and generic object recognition. In object recognition, the images or videos have high resolution, in the megapixel range, while traffic monitoring cameras often provide low resolution videos in the kilopixel range. Hence, it becomes difficult to extract features that rely in specific details of the vehicles. Thus, in traffic monitoring systems usually it is given less importance to vehicle categorization and instead they try to provide a robust system capable of detection of false vehicles, in poor quality videos, based on the extracted features.

Therefore, in traffic monitoring systems the extracted features are mainly used to identify objects as candidate vehicles. Hence, it uses features of the object such as their position, their size or their distance to each other to validate the objects as vehicles, as illustrated in Fig.8, and thus reject other objects resulting from noise or double detections of the same vehicle. This allows a higher level of robustness to the system, especially in the presence of harsh operation conditions as inclement weather.
2.3.4 Video-based Traffic Management Systems

Video cameras have already been deployed for years for traffic monitoring purposes, because they provide a rich information source for traffic, which is easy for a human operator to understand. Video-based Traffic Management Systems try to replace tedious and repetitive tasks related with counting the vehicles in the cameras and classifying the traffic according to it and instead give the processed information directly to the human operator. The following sub-sections discussed three distinct systems used to acquire traffic information from videos.

2.3.4.1 Automatic Number Plate Recognition

Automatic Number Plate Recognition (ANPR) systems are specialized systems used for vehicle identification. The cameras used are highly zoomed to provide an image with high resolution of the vehicle and its number plate, allowing its identification. This system is extensively used in highways, which have dedicated cameras placed in the toll stations to enter and to exit the highway, which identify the plate number of the vehicle and communicate it to a data center. With the information of where the vehicle entered and exited the highway, it is possible two know how much it has to pay. Since the ANPR provides information of the vehicles entering and exiting the highway, it is also possible to analyze the traffic on those roads. Such a system can accurately estimate the number of vehicles on the highway as they have to always cross the toll. However, it is hard to identify precise areas of possible congestion inside the highway as the information available is just at the entrance and exit of the road. In Fig.9, it is possible to see the cameras on the upper side of the toll stations using ANPR for tolling.
Fig. 9: Example of toll station with ANPR [Brisa, 2014]

2.3.4.2 Virtual Detection Lines

The Virtual Detection Line (VDL) based system uses a VDL to assess the traffic on road lanes [Mithun, 2012]. This system simply places a VDL across the lanes in which the traffic should be analyzed and counts the vehicles crossing it.

In order to count the vehicles, the system generates a Temporal-Spatial Image (TSI) using the pixels of the frame of the video which were placed on the drawn VDL. Subsequently, the background of this TSI is estimated using a background model allowing to compute a mask with blobs representing moving objects, with each of this objects being called a TSI object blob (TOB). These TOBs then go through a process of classification, using features extracted from these TOBs, evaluating if the object is really a vehicle and thus avoid false positive classification of vehicles. With the vehicles identified, it is possible to assess the traffic in the chosen lanes. It is a simple system that presents low execution times and high accuracy. However, the correct placement of the VDL perpendicular to the lanes has a strong effect on the results obtained. Examples of VDLs, TSIs and TOBs are illustrated in Fig. 10.

Fig. 10: Left: VDL in sample image; Centre: Resultant TSI; Right: Resultant TOB

2.3.4.3 Thermal Imaging

A monitoring system developed by FLIR [FLIR, 2014] uses thermal imaging provided by special thermal cameras, developed by the same company, to analyze the traffic and thus provide information such as traffic management and congestions, accidents or dangerous pedestrians’ behavior on the
roads, allowing warning signals to be activated in these situations. These thermal images are illustrated in Fig.11.

The advantage of the thermal cameras is that they are not especially affected by low luminance detection problems, working in situations with no light at all according to the company, which allows good detection of objects even in poor lighting conditions. These cameras can also work in different weather conditions. However, these thermal cameras would have to replace the usual LTSV cameras used for traffic management, which may turn them a pricey option to implement.

![Thermal image provided by a thermal camera](image)

**2.4 Operational Conditions**

As stated before, traffic monitoring systems usually have low resolution cameras, for instance, most of the videos used in this dissertation have a resolution around 200x200, and thus a limited amount of visual detail on the captured scenes. Hence, taking the noise into account, the quality of the data is generally poor, which means that the systems used have to be robust as they are going to be used in distinct operational conditions which include challenging weather conditions and night time. The following sections discuss the common problems faced by these systems and, when available, suggest some of the solutions used in traffic management systems reported in the literature. These problems include:

- Occlusion – addressed in section 2.4.1;
- Night time videos – addressed in section 2.4.2;
- Weather – addressed in section 2.4.3;
- Shadows – addressed in section 2.4.4.

**2.4.1 Occlusion**

Occlusion occurs when the view of an object is partially or completely obstructed by another. In the cameras used in traffic management systems, this is usually caused by the high traffic density on the road or by a poor angle of the camera placement. These situations are illustrated in Fig.12, where in the first image most vehicles are being occluded by other vehicles, and in the second image the camera is pointing at an information sign over the road, which is partially blocking the view of the passing vehicles, hindering the traffic count. As a result, vehicles might be missed, reducing the overall system performance. To avoid these situations in a traffic management system, the cameras...
should be placed in high poles over the road, which provide a better view angle of the road and the vehicles [Buch, 2011].

![Fig.12: Example of occlusion in a LTSV camera](image)

This section presents methods used to attenuate this effect. Hence, the following sub-sections explain three methods which can be deployed on traffic management systems to tackle this problem, followed by a comparison of their performances.

### 2.4.1.1 Backtrack-chain-update split algorithm

The method proposed by [Chen, 2001] works well to split partially occluded objects with similar sizes and shapes that move on different directions, for instance, turning on a crossroad. It is useful in cases when the camera is not positioned high enough to see the vehicles from the top but instead almost from the front as, for example, in a traffic light.

The objects are tracked by their centroids. Size restrictions are applied to the object, taking into account the frame rate of the camera, since the size should not change considerably between two frames. The previous frame is used as an initial condition. If two overlapped segments separate from each other between frames, the previous frame may not find its corresponding part in the current frame due to changes in the size of the object or in the position of the object. Therefore, some segments cannot be tracked back to the previous frame, these unidentified segments are analyzed to find the relationship between them on the previous picture using the following algorithm:

The first step of the algorithm starts by finding all the related segments between both frames identifying them as “parent” and “child”.

The second step identifies the split between objects in the current frame and relates them to the previous picture building the parent-child relationship.

The third step does a segmentation of the next frame and builds the parent-child relationship. According to the size of the split segments on frame \( i + 1 \) related to their size in frame \( i \), a sensitivity parameter for their width and length is created taking into account the frame rate of the camera.

The fourth and last step backtracks and updates the previous frame with the information of the split segments. To that effect, a recovery vector is created using the split segments in frame \( i + 1 \) and frame \( i \). This vector is used to guess the split segments in frame \( i - 1 \).
Chen’s method has some limitations. It needs the objects to have the same size and shape, they cannot change considerably during consecutive frames and therefore the algorithm’s parameters are sensitive to the frame rate. Furthermore, it is made to detect the occlusion when the vehicles start to follow different paths. Therefore, it won’t work efficiently if they keep moving in the same lane.

2.4.1.2 Simultaneous Partition and Class Parameter Estimation

This method follows the same general idea of the previous one, while trying to overcome the identified limitations concerning size and shape of objects.

The Simultaneous Partition and Class Parameter Estimation (SPCPE) algorithm is a stochastic model-based unsupervised video segmentation method to partition video frames. It considers the image classes as random fields and the segmentation problem as a statistical optimization problem [Sista, 2000]. The method for partitioning a video begins with an arbitrary partition followed by an iterative process to estimate the class parameters jointly. Each new frame is partitioned taking the previous frame partition as an initial condition. For the initial frames a random initial partition is generated [Chen, 2001].

Considering two classes, let the partition variable be $c = \{c_1, c_2\}$ and the classes be parameterized by $\theta = \{\theta_1, \theta_2\}$. Let all the pixel values $y_{ij}$ of an image $Y$ belonging to class $k$, be put in a vector $\mathbf{y}_k$. Let each row of a matrix $\phi$ be given by $(1, i, k, ij)$ and $a_k$ is a vector of parameters $(a_{k0}, \ldots, a_{k3})^T$. We compute the following equations to have a estimation of the parameters $\hat{a}_k$:

\[
y_{ij} = a_{k0} + a_{k1} i + a_{k2} j + a_{k3} ij, \forall (i, j)
\]

\[
\mathbf{y}_k = \phi a_k
\]

\[
\hat{a}_k = (\phi^T \phi)^{-1} \phi^T \mathbf{y}_k
\]

To find the best partition it is necessary to maximize the a posteriori probability (MAP) of the partition as variable given by the image $Y$. Therefore to estimate $c = \{c_1, c_2\}$ and $\theta = \{\theta_1, \theta_2\}$, we have:

\[
(\hat{c}, \hat{\theta}) = \text{Arg} \max_{(c, \theta)} (c, \theta | Y)
\]

Letting $J(c, \theta)$ be the cost function to be minimized, it can be written as:

\[
(\hat{c}, \hat{\theta}) = \text{Arg} \min_{(c, \theta)} J(c_1, c_2, \theta_1, \theta_2)
\]
\[ J(c_1, c_2, e_1, e_2) = \sum_{y_{ij} \in c_1} -\ln(p_1(y_{ij}; e_1)) + \sum_{y_{ij} \in c_2} -\ln(p_2(y_{ij}; e_2)) \]

Therefore, from the starting arbitrary partition of the data, the class parameters are computed. These class parameters and data are afterwards used to estimate a new partition of the data. New class parameters and data partitions are iteratively refined until there is no significant change. The refined partitions are afterwards individualized by a minimal bounding rectangle (MBR), as illustrated in Fig.13.

This approach can effectively track objects that undergo rotations or translations. However, SPCPE cannot partition overlapping objects efficiently by itself and to correctly identify them, hence it needs some enhancements [Chen, 2001].

The method proposed by [Liu, 2011] uses an n-step search to tackle this problem. It is composed of a loop of four steps which are iterated for all frames.

On a first step, it starts by estimating the background and foreground of a frame \(i\) with the help of an unsupervised SPCPE based on the background and foreground of frame \(i-1\) as an initial partition. Then the background of \(i\) is removed using the estimation and bounding boxes are placed around the foreground.

On a second step, the object occlusion is detected by using the size and information of bounding boxes in two consecutive frames. If there is an object occlusion situation, the bounding box of the occluded objects is passed to the third step, otherwise the loop goes back to the first step or ends if it is the last frame.

On a third step, to identify the location of the occluded objects an n-steps search (NSS) method is used by retrieving the spatial information of the objects in frame \(i-1\), and thus the preliminary detection results of frame \(i\) are generated.

On a fourth step, a size-adjustment method is developed to adjust the bounding boxes of the occluded objects to obtain more accurate sizes and positions of these objects, returning to the first step, unless it is the last frame.

Fig.13: Left: result of SPCPE; centre: result of NSS; right: result of size-adjustment [Liu, 2011]

This method shows good results especially when the occluded vehicle moves in a different direction in relation to the occluding object, however it is not as efficient when both vehicles follow the
same path.

2.4.1.3 Multiple Virtual Detection Line-based Detector

As already discussed in section 2.3.4.2, VDL is one of the possibilities available to be used in traffic management systems. However, when vehicles are moving too close to each other, it may fail to detect some of them, especially in darker scenarios, due to occlusion in the TSI. One possible approach to solve the occlusion effect is the use of Multiple Virtual Detection Lines (MVDLs).

The method presented in [Mithun, 2012], uses MVDLs to detect cases of occlusion. To that purpose, MVDLs are positioned in the same lane parallel to each other at certain predetermined distances. The resulting TSIs are transformed into binary masks with the help of an edge detector such as the ones discussed in 2.3.2. The resulting blobs of this process are then used by an algorithm which uses a matrix $N$, of size $n \times m$ and a array $M$ of size $n$, where $n$ is the number of blobs and $m$ is the number of vehicles. The matrix $N$ keeps trace of whether the blobs of the distinct vehicles are disjoint or merged, while the array $M$ keeps track of the position of each blobs.

To detect occlusion the algorithm selects the left most blob of one TSI and searches on the other TSIs as is illustrated in Fig.14. The occlusion is detected computing the dissimilarity between the same blob identified in different TSIs finding the logical values of the following equations $\gamma_1$, $\gamma_2$, $\gamma_3$ and $\gamma_4$:

\[
\gamma_1: \frac{|C^y_\ell - C^y_n|}{C^y_n} > T^y_C
\]

\[
\gamma_2: \frac{|W^y_\ell - W^y_n|}{W^y_n} > T^y_W
\]

\[
\gamma_3: \frac{|W^x_\ell - W^x_n|}{W^x_n} > T^x_W
\]

\[
\gamma_4: \frac{|A_\ell - A_n|}{A_n} > T_A
\]

where:

- $\ell$ and $n$ are the indexes of the associated blobs in $TSI_\ell$ and $TSI_n$;
- $C^y_n$ is the value of the coordinate $y$ of the centroid of the blob $n$;
- $W^y_n$ and $W^x_n$ are the height and width of the blob $n$;
- $A_n$ is the area of blob $n$;
- $T$ are the thresholds.

The thresholds were chosen between 5% and 15% and may be dependent of illumination,
vehicle size, and video resolution. Occlusions are detected when the condition \( i \) in (2.26) is satisfied:

\[
\begin{align*}
\hat{i} & : f_1 \land (f_2 \lor f_3) \land f_4
\end{align*}
\]

In case of detected occlusion, the number of merged objects is computed and then \( N \) and \( M \) are updated accordingly, otherwise the algorithm moves for the next blob of the first selected TSI. The process is repeated until all the blobs are verified on each TSI.

This method works correctly for grayscale and low resolution videos. However, the vehicles should not change lanes to be tracked and compared correctly between all the TSIs.

![Diagram](image)

Fig. 14: Section of scaled version of multiple TSIs generated from MVDLs [Mithun, 2012]

2.4.1.4 Performance comparison

The computational and memory complexity of the methods discussed are proportional to the number of frames used. Therefore, when discussing the performance of these methods, we are referring to both complexities simultaneously.

The methods in 2.4.1.1 and 2.4.1.2 have very similar complexities. The main difference is that the first one needs 3 frames to compare and establish the relations between the vehicles in the video and the second needs just 2. The MVDLs method has a complexity dependent on the number of VDLs used. If the method uses only 2 or 3 VDLs, it will have a complexity similar to the other two presented methods. However, if more VDLs are used, its complexity will increase proportionally to the number of VDLs.
2.4.2 Night time

The detection methods used to find vehicles in traffic management systems are always related with the intensity of the pixels provided by the frames of the videos, since they use the variations of intensity on these pixels to detect the object in these images. The variation of the pixels at night is much lower due to the low amount of luminance existent in the scene, as illustrated in Fig.15, compared to the luminance during day time with clean weather.

Fig.15: Difference between night time and day time images, with clean weather

There is very limited literature in vehicle monitoring specialized with night time and difficult light [Buch, 2011]. Therefore, in night time it is used generally the light provided by the headlights or taillight as the features to detect vehicles in the videos. Hence, traffic management systems have to adjust the parameters it uses in its detection phase to acquire these objects instead of the usual structure of the vehicle and thus the characteristics used to search for possible false vehicles on the classification phase also have to be adjusted.

2.4.3 Weather

Distinct types of weather provide different challenges to a traffic monitoring system. Rainy weather, as illustrated in Fig.16, may cause puddles on the roads which cause reflection in the video’s image and may cause the detection of false vehicles.

Fig.16: Difference between rainy and clean weather images

The visibility of the traffic from the cameras tends to be reduced, being specially noted on foggy weather. This reduced visibility is of special importance because it hinders the vehicle detection, once again, due to the reduction of the pixels intensities which cause a lower variation between them and
reduce of efficiency of the object detection phase. Once more, the related literature doesn’t provide significant specialized research in these situations [Buch, 2011]. To mitigate the lower efficiency cause by lower variations between pixels, the detection methods tend to be adjusted with higher sensitivity to pixels changes. These changes usually are made by lowering thresholds or reducing the size of smoothing filters used to reduce the noise, such as the Gaussian Filter.

2.4.4 Shadows

The shadows of moving objects need a careful consideration in traffic management systems. The detection of moving shadows affects the efficiency of the system as they are often misclassified as moving objects when performing the foreground estimation and may cause errors in the assessment of the traffic. This especially true when the shadow of a vehicle overlaps other vehicle, as illustrated in Fig.17, as it may cause both vehicles to be considered as just one in the object detection phase.

Fig.17: Left: VDL on sample image; Right: Resultant TSI with object detection showing occlusion

The methods used in the literature to deal with shadows are divided in two types, property-based and model-based. On one hand, property-based methods use extracted features from the objects, such as geometry, color or brightness, to identify the shadow regions of the objects. On the other hand, model-based methods use a priori knowledge of the scene geometry, the foreground objects or the light sources to detect the presence of moving shadows on the objects.

In management systems, due to the usually poor quality of the video data, the methods used tend to be model-based, since knowing that shadows may be occurring on the video, allows the system to be adapted, adjusting its detections parameters and filters used, and thus minimize the effect of the shadows on the detection of the objects.
Chapter 3

Proposed System

This chapter details the components and procedures applied to the traffic management system developed in this dissertation.
3.1 Overview

The system proposed in this dissertation, as mentioned before, is a Video-based Traffic Management System, which uses an image processing algorithm to estimate the number of vehicles from uncontrolled traffic monitoring video cameras. The system has to be robust enough to handle distinct operational conditions, such as night time or inclement weather, and also be able to minimize the number of counting errors due to occlusions.

The video sequences employed for testing the system, and used for illustration purposes in the following sections, were acquired from uncontrolled LTcv cameras, courtesy of Brisa, Estradas de Portugal and Youtube. They have low resolution, usually around 200x200 pixels or less, and the acquiring cameras are installed far from the road lanes, thus the visualized vehicles mostly appear as small objects with a minimum amount of detail. Therefore, the methods used on this work had to rely on object features such as area, position and moving direction to decide if it should be considered as a possible vehicle. These features have to be extracted from the TSI generated from the provided video sequence of the camera according to the chosen VDLs of the road lane.

The system thus follows a sequence of steps to acquire the features that allow it to obtain a vehicles list, as illustrated in Fig.18. First, the user should select the road lanes in which the vehicles are to be counted, by drawing one or more VDLs crossing it. Afterwards, the system generates TSI for the corresponding VDLs. It is followed by the background estimation of this image and subsequent subtraction of the background from it. Then the contours of the objects accentuated by the subtraction are drawn using a Canny Edge Detector. Finally, each object is separately identified and its features acquired. This is followed by a False Vehicle Elimination (FVE) step where the object features, such as area and position, are used so that some of these objects can be rejected as possible vehicles. The final step uses a second VDL of the same lane to verify possible vehicles occluded by other objects and computes the number of vehicle vehicles from the analyzed objects.

The following sections provide a detailed explanation of what is done in each step of the system. Then, a final section describes the graphical user interface developed to allow the user to interact with the system.
Fig. 18: Diagram with a summary of each step of the system
3.2 VDL Selection

The proposed system starts by allowing the user to define a set of VDLs crossing the lanes that should be analyzed. This can be done in two distinct ways:

- by writing the coordinates of the two endpoints between which the line should be drawn;
- clicking in a sample image provided by the image sequence of the camera to select the desired endpoints.

The resulting VDL is generated using Bresenham’s line algorithm, which determines which pixels should be plotted in the image to generate a close approximation of a straight line between two points [Wright, 1990].

The general equation to find the pixels which belong to the line is given by equation 3.1.

\[
\frac{y - y_0}{y_1 - y_0} = \frac{x - x_0}{x_1 - x_0}
\]

where:

- \((x, y)\) represents the location of a line’s pixel in an image;
- \((x_0, y_0)\) and \((x_1, y_1)\) represent the endpoints of the line;

Bresenham’s line algorithm takes one endpoint and starts generating the line’s pixels in direction to the other endpoint, as illustrated by Fig.19. Hence, if the process is started from \((x_0, y_0)\) to \((x_1, y_1)\) and \(x_0 < x_1\), then at every new point generated the x value is incremented, \(x = x_0 + 1\). Knowing x, the value of y can be easily found afterwards by manipulating the previous equation 3.1 to:

\[
y = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) + y_0
\]

Fig.19: Bresenham's line algorithm example [Crotalus, 2014]
The next example is shown in Fig.20 where three VDLs are drawn using the mouse’s cursor, nonetheless the coordinates are displayed above the figure in the four text input boxes for the \((x, y)\) coordinates of the two endpoints needed to generate a VDL, which can be adjusted. The complete description of the graphical user interface (GUI) of the proposed system is presented in Annex 1-Graphical Interface.

![Fig.20: Interface to define the VDLs by the user.](image)

### 3.3 TSI Generation

The TSIs provide the core information for the whole processing system. They are generated accordingly to the VDL position drawn by the user over the provided sample image of the video sequence. The steps of the process are illustrated in the flowchart of Fig.21.
Fig. 21: Flowchart of the TSI Generation

First, the vector of pixels extracted of each frame of the video sequence provided by the camera must be saved onto a matrix according to 3.3:

\[ m_{jk} = v^k_j, \quad \forall k \in A \]  \hspace{1cm} 3.3

where:

- \( A \) is the set of frames belonging to the video sequence
- \( I_k \) is the frame \( k \) of the video sequence;
- \( v^k_j \) is a vector of size \( J \) with the intensities of pixels of the video which overlap the VDL in frame \( k \);
- \( m_{kj} \) is a matrix of size \( J \times K \) where the acquired pixels are stored;
The pixels belonging to the VDL on each frame are saved on $v^k_j$ and sorted from the top row to the bottom row, as illustrated in Fig.22. For pixels with the same y value, as for instance pixels 1 and 2 of Fig.22, the pixels are sorted from the left to the right. Hence, the first pixel saved in the vector is always the pixel belonging to the endpoint with lower value on the x axis and the last pixel saved in the vector is the other endpoint.

![Fig.22: Pixel assortment in the vector $v^k_j$](image)

The size of each TSI will depend on the duration of the video used and the length of the VDL drawn, as shown in Fig.23. The TSI will extend vertically according to the length of the line and horizontally depending on the video’s stream duration.

![Fig.23: Left: VDL in video’s sample; Right: Corresponding TSI according to its length and video’s time.](image)

If the video used to generate the TSI has several color channels, an extra step will convert it to a single grayscale channel to simplify the subsequent background estimation and subtraction.
3.4 Background Estimation & Subtraction

The background estimation and subtraction is necessary to know what is not moving in the video and hence should be ignored during the TSI analysis for object detection. The estimation method being proposed in this dissertation tries to select a better background in comparison to the more commonly used alternative methods for background estimation on TSIs. A comparison with alternative methods is done at the end of this section. The process is summarized in Fig. 24 and described in the following paragraphs.

![Flowchart of background estimation and subtraction](image)

**Fig. 24: Flowchart of background estimation and subtraction**

The proposed method assumes that the column most representative of the background in the TSI can be selected as a background column to be subtracted from the whole TSI image to help in the detection of the vehicles passing by.

1. **TSI pixel intensity histogram**

The pixel intensity histogram is performed by:

\[
h(z) = \sum_{i=1}^{N} \sum_{j=1}^{M} 1, \forall I(i,j) = z, \text{where } z \in \mathbb{Z}
\]

where:

- \( I(x,y) \) is a \( N \times M \) matrix with the pixel values of the TSI;
- \( h(z) \) is a function dependent on a parameter \( z \in \mathbb{Z} \).

We obtain \( h(z) \) as a histogram of the values of the TSI \( I(x,y) \).

2. **Column selection**

Each pixel of the TSI \( I(x,y) \) receives a distinct score according to its frequency on the whole TSI, which is performed by:
where $V_i(x, y)$ is a matrix of size $N \times M$ with the scores of all the pixels.

We obtain a matrix with the size of the TSI with each element's score reflecting the frequency of occurrence of that particular pixel intensity in the whole TSI.

In order to acquire the sum of the scores on each column the sum of each column of $V_i(x, y)$ is computed, which is defined by:

$$v_x = \sum_{i=1}^{M} V_i(x, i)$$

Where $v_x$ is a vector of size $N$.

Finally, the column of $V_i$ with highest score and its index can be computed by:

$$v^*_x = \max\{v_x\}, \forall x \in \mathbb{Z}$$

where $v^*_x$ is the highest value of $v_x$ and has the index values.

The column with index $s$ has the highest score and thus it is the TSI column, which has the most frequent pixel values in the whole TSI. Hence, it has the highest probability of being a good representation of the background of the TSI.

### 3. Column subtraction

Once the index of the column with the highest score is known, it is selected as the TSI background model, $b_s(y)$, and can subsequently be subtracted in the whole TSI. There is only one selected index instead of distinct indexes over the length of the TSI due to the use of only a few seconds of video, usually no more than 20 seconds, to generate each TSI. Therefore, there is no significant background change in this amount of time. The TSI background is thus subtracted computing:

$$S(i, y) = |b_i(y) - b_s(y)|, \quad \forall i \in M$$

where:

- $S$ is a $N \times M$ matrix with the foreground estimation of the TSI;
- $b_i(y)$ be the column $i$ of $I(x, y)$;
- $b_s(y)$ the column with the index $s$ of $I(x, y)$.

We obtain the TSI subtraction leaving just the module of the difference of the pixels as a result. The remaining value in the pixels ideally represent the foreground in the image, which should
contain the moving vehicles to be found by the edge detection step.

3.4.1 Comparison with usual column selection method

There are other alternative ways to choose the column to be used for subtraction on the TSI. Hence, the following paragraph discusses the differences in selection and the results between the method proposed in this dissertation and an alternative proposed in [Santos, 2012].

In [Santos, 2012] the background estimation of an image is made using the TSI column exhibiting the smallest pixel variance. That column is selected for the background subtraction. This approach assumes that the background will be a road and thus the variance of the pixels should be minimal. However, this is not always true. When there are, for instance, puddles of water causing light reflection due to rain, or a shadow affecting part of the road caused by a tree or a building, the minimum variance won’t choose the best column. It may choose a line over a dark vehicle with a color close to that of the road or any column which displays a large object with a uniform color, which will present a lower variance than the actual road surface. Examples of distinct results obtained by the two methodologies are illustrated in Fig.25.

![Fig.25: Left: VDL used for illustration of background subtraction operation; Center: proposed method; Right: minimum variance method](image)

Fig.25 shows that using the minimum variance method the edge detection will find extra contour lines along the lane, motivated by not being able to select the best TSI column for the background subtraction operation. These lines occur in a TSI area where there is some light reflection caused by water in the pavement. This reflection causes the pixel values to be higher than the rest of the lane and thus the column chosen by the minimum variation method is quite different than the one used here. These extra lines make it difficult or impossible to count vehicles in that particular lane due to the connection that the contours of the vehicles have with these undesired contour lines. The proposed method is able to overcome the identified problem due to the importance given to the frequencies of pixels with a certain intensity in a certain column, instead of their variation. This way, it is possible to obtain a better column for subtraction in situations where the minimum variance would select an incorrect column, while both obtain similar results in the cases where the minimum variance works correctly.
3.5 Edge & Object detection

The edge and object detection tries to find the moving objects on the TSI. It works on the modified TSI, which is the result of the background subtraction of the previous step. This modified TSI has the moving foreground objects which includes the moving vehicles on the road. This section describes the process to find and individualize the foreground objects.

This section is mainly divided into two parts:

- An edge detection step to identify the contours of the objects of the TSI;
- An object detection step to individualize the objects corresponding to the previously identified contours.

It should be noted that on night scenes, an extra step should be performed according to the camera angle to the road. In case the vehicles are moving towards the camera, an Otsu's binarization is performed to highlight the reflection of vehicles' headlights on the road pavement, and it is used to detect vehicles instead of their body structure. In case the vehicles are moving away from the camera, a slight modification of the TSI is done to reduce double detections due, once again, to the reflection of vehicles' headlights on the road pavement. However, this time the modification is done in order to ignore the headlights' reflection. The night situations where this step is used are further explained on section 3.11.

The flowchart detailing to the process of edge and object detection is included in Fig.26.

1. Edge detection

The contours of the moving objects obtained after background subtraction are searched for with the help of a canny edge detector. This process takes four steps as described in section 2.3.2.2. The values used on this process are detailed on the following paragraphs.

The aperture value used in the Canny detector changes the aperture size of the Sobel
operator. The options available to the size of the aperture are 1, 3, 5 or 7. The first option tends to be too insensitive to detect the contour of the objects in the TSI, while the last option detects every detail of the TSI as a contour, acquiring a high amount of noise as objects, becoming unpractical. Hence, the only practical choices are 3 or 5. During the experiments performed, having an aperture of size 5 proved to be more stable than 3. It was able to give the same results when the Canny threshold value changed slightly, which is an especially important characteristic when the system is working with automatic parameters. On the other hand, with an aperture of size 3 small changes on the Canny threshold values caused quite distinct vehicle counts. Therefore, an aperture size of 5 was assigned to the Canny edge detector.

The Gaussian filter is used to clear the noise on the TSI. However it will also erase some of the information present on the TSI. Since the video sequences of traffic surveillance cameras usually have low resolution, the generated TSIs also have low resolutions of the objects. So even small filter sizes might destroy important information needed to individualize the moving objects. In order to clear some of the noise and still be able to detect the vehicles a filter size of 3 is used on the developed system. Higher values tend to increase the number of occluded vehicles on the TSI as the contours individualizing them blend together due to the filter. Furthermore, high values tend to also decrease the detection of darker vehicle due to their low pixel intensity contrast relative to the road pavement. Lower values tend to leave the TSI with a significant number of small objects which is just noise contained on the video sequence, which might decrease the vehicle count accuracy.

The gradient is computed using the 5x5 Sobel convolution masks $G_x$ and $G_y$ as follows:

$$G_x = \begin{bmatrix} 1 & 2 & 0 & -2 & -1 \\ 4 & 8 & 0 & -8 & -4 \\ 6 & 12 & 0 & -12 & -6 \\ 4 & 8 & 0 & -8 & -4 \\ 1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 1 & 2 & 6 & 2 & 1 \\ 4 & 8 & 12 & 8 & 4 \\ 0 & 0 & 0 & 0 & 0 \\ -4 & -8 & -12 & -8 & -4 \\ -1 & -2 & -6 & -2 & -1 \end{bmatrix}$$

The gradient strength and direction are then computed using:

$$G = \sqrt{G_x^2 + G_y^2}, \quad \theta = \arctan\left(\frac{G_y}{G_x}\right)$$

The directions are discretized into one of four directions, namely 0, 45, 90 and 135 degrees, being $\theta$ rounded to the closest of these values.

Finally, the low threshold and the high threshold for the gradient strength are used to generate the contours. The low threshold defines the gradient strength of the values which are automatically rejected by the detector, while the high threshold defines the values which are always accepted as contours. The values between the thresholds are accepted if they are neighboring an already accepted value. The best results with the lowest detection of undesired objects caused by noise, occurred when the difference between the low threshold and the high threshold had a ratio in the range between 1:2 and 1:3. Lower threshold ratios tend to detect more noise as moving objects, while
high threshold ratios tend to not detect the darker vehicles as they have pixel values close to the intensity of the road lane. The values used in the thresholds vary according to the illumination on the scene and are further discussed in section 3.11.2.1.

2. Object detection

The object detection is made with an algorithm developed presented in [Suzuki, 1983]. This algorithm uses a border following procedure to perform a topological analysis of the provided binary image of the edge detector. It assigns a value of 1 to outer borders of the contours and a value of 0 to any other pixel. The algorithm just has to follow the 1-components neighboring each other to individualize each contour. The algorithm has the option to individualize all the contours or just the outer contours, in case there are contours inside of other contours. Since in a traffic management system just the outer contours are relevant, the second option is used to individualize the contours.

3.6 Feature Extraction

The feature extraction goal is to acquire features from the objects found on by the edge and object detection steps in order to have more information about these objects. This information is essential to be able distinguish the objects that are vehicles and the ones that are not and thus must be rejected. Therefore, this section describes the process followed to acquire those features, which consists of three steps as illustrated in Fig.27.

![Flowchart of Feature Extraction](image)

**Fig.27: Flowchart of Feature Extraction**

1. Generation of bounding rectangle

In order to obtain this information, it is necessary to use the generated contours of the previous step. A bounding rectangle is created around the contours with the minimal width and height. The rectangle’s area is always an upper bound of the area of the contour used to generate it. An example of result obtained with this procedure is illustrated in Fig.28 for a sample TSI.

![Results](image)

**Fig.28: Left: Contours detected; Right: TSI with bounding rectangles with highlighted centroids.**
2. Estimation of centroid

The centroid of the contour is estimated using the width and height of this rectangle, according to the following expression 3.11:

\[
\begin{align*}
    x_i &= \frac{(x_{br} - x_{tl})}{2} + x_{tl} \\
    y_i &= \frac{(y_{br} - y_{tl})}{2} + y_{tl}
\end{align*}
\]

where:
- \((x_{tl}, y_{tl})\) is the top left point of a rectangle;
- \((x_{br}, y_{br})\) is the bottom right point of the rectangles;
- \((x_c, y_c)\) is the estimated centroid of the object.

The centroids are used to have the approximate position of each detected object, as it is necessary in the following steps of the system.

3. Acquisition of objects’ features

A set of simple and fast to compute features are thus extracted in this step, notably consisting in estimates of the width, height and centroid of the contours previously detected, using the created bounding rectangle.

3.7 False Vehicle Elimination

There are situations where undesired objects are identified causing false vehicle detections. These spurious objects usually appear due to noise or double detections of the same vehicle. In order to eliminate these undesired objects, a verification of the corresponding features, namely their area, width and height, allow deciding whether to keep them. The method proposed and developed in this dissertation for False Vehicle Elimination (FVE) is composed of three processes for vehicle validation, as described in the following paragraphs and illustrated in Fig.29.
1. Verification of objects’ bounding box edges

The objective of the first proposed process is to eliminate the objects which have small and unusual dimensions, namely a small width and an extremely long height or a small height and an extremely long width. These objects are noise that reduces the vehicle counting accuracy. To do so, the width and height of the objects are analyzed. When equation 3.12 is satisfied, the object is eliminated. Otherwise the object is kept.

\[
\left( \text{height}_i < \delta_h \land \text{width}_i < \frac{\text{width}_i}{\text{height}_i} \right) \lor \left( \text{width}_i < \delta_w \land \text{height}_i < \frac{\text{height}_i}{\text{width}_i} \right)
\]

where:
- \( \text{width}_i \) is the width of the object \( i \) in pixels;
- \( \text{height}_i \) is the height of object \( i \) in pixels;
- \( \delta_h \) and \( \delta_w \) are values for minimum height and width of the object in the interval;
- \( r \) is a ratio between the width and the height of the object.

The values for \( \delta_h, \delta_w \) and \( r \) can be adjusted by the user. By default, \( r \) is set to 3, \( \delta_h \) and \( \delta_w \) are
set to 4. These values are constant independently of the distance of the vehicles to the camera as this type of noise may be present in any video. Higher values for the minimums and ratios tend to reject vehicles when the cameras are far away from the targeted road lane for analysis. An example of the small edge verification output is included in Fig.30. The blue rectangles on the figure are detections of light reflections on the pavement which produce false positive detections. The use of the small edge verification algorithm allowed an automatic rejection of those candidate objects.

![Fig.30: Left: TSI without edges verification; right: TSI with edges verification](image)

2. Verification of objects’ area

The second developed process is used to verify the objects’ area using equation 3.13. It is performed to reject objects with a small overall area. When the equation is satisfied, the object is rejected. Otherwise the object is not rejected.

\[
\text{area}_{\text{object}\ i} < \alpha \cdot \text{area}_{\text{average}} , \quad \forall i \in X
\]

where:
- \(X\) is the set of objects detected;
- \(\text{area}_{\text{object}\ i}\) is the area of the object \(i\);
- \(\text{area}_{\text{average}}\) is the computed average area of the objects detected;
- \(\alpha\) is an arbitrary value chosen by the user in the interval \([0,1]\).

When the objects have an area below the selected threshold, \(\alpha \cdot \text{area}_{\text{average}}\), dependent on the average area, the object is considered noise and thus rejected.

In bright scenes, this method typically presents good results for values of \(\alpha\) around 0.25. Higher values sometimes cause the rejection of smaller vehicles, such as motorbikes. In Fig.32, there is an example of objects removal due to their small area, which is highlighted with black rectangles.

In dark scenes, there is a higher probability of noise being detected as objects due to the higher sensitivity of the Canny threshold generally used. This reduces the system counting accuracy. To tackle this issue, in these scenes the value of \(\alpha\) is increased to 0.5 to reject those undesired objects. An example of the distinct result obtained while using 0.25 and 0.5 for \(\alpha\) at night is included in Fig.31.
3. Verification of overlapping rectangles

The third verification introduced in the FVE developed in this dissertation searches for the duplication of objects for the same vehicle, due to multiple detections of the same vehicle. The multiple detections can usually be removed by eliminating the detected objects which are at least partially overlapped. The strategy used to tackle this is based on the comparison of the overlapping area, with the total area of each of the individual rectangles. When this overlapping area is larger than a given threshold, the rectangle is rejected. However, both rectangles cannot be simultaneously rejected, hence when a rectangle is rejected the one with a bigger area is always kept. The following paragraphs explain the whole process.

The position of all the objects is verified with the help of their centroids position and if the centroid of one of the objects is inside the bounding rectangle of another object, the objects overlapping area can be computed using expressions 3.14, 3.15 and 3.16.

\[
\text{width} = \min(x_a + \text{width}_a, x_b + \text{width}_b) - \max(x_a, x_b) \quad 3.14
\]

\[
\text{height} = \min(y_a + \text{height}_a, y_b + \text{height}_b) - \max(y_a, y_b) \quad 3.15
\]

\[
\text{area}_s = \max(0, \text{width}). \max(0, \text{height}) \quad 3.16
\]

where:

- The rectangle \( a \) and \( b \) are the bounding rectangles of the two objects being analyzed, sorted by increasing size;
- \((x_a, y_a)\) and \((x_b, y_b)\) are the top left corner location of \( a \) and \( b \);
- \( \text{area}_s \) is the overlapping area.

Considering that we are verifying if the rectangles should be rejected, letting \( \text{area}_a \) be the area of rectangle \( a \), which is the rectangle with smaller area, and \( \beta \) an arbitrary value chosen by the user in the interval\([0,1]\), we have:

\[
\text{area}_s > \text{area}_a \cdot \beta \quad 3.17
\]

The overlapped area, \( \text{area}_s \), is thus compared with the \( \text{area}_a \). When 3.17 is satisfied, the
common area is superior to a percentage of the area of this rectangle $a$, and thus object $a$ is rejected, otherwise there is no rejection. Rectangle $b$ is never rejected as it is assumed that the bigger rectangle contains the main body of the vehicle and the smaller rectangle may be other vehicle near vehicle $b$ or a detail of the vehicle of $b$, which in this case must be ignored in the counting.

In bright scenes, this method presents the best results when $\beta$ is defined with a value superior to 0.5 of the area of the object being checked. Lower values may cause the rejection of vehicles moving too close to each other, eventually partly occluding one another. Hence, in bright scenes this value is set to 0.65. In Fig.32, there is an example of objects’ removal due to their overlapping area, which is highlighted with yellow rectangles.

![Fig.32: Left: Vehicle detection result without and with FVE, respectively on left and right.](image)

In dark scenes, the number of double detections of the same vehicle tends to increase. This may be caused by the detection of the vehicle and the headlights of the vehicle as two distinct objects or due to the detection of each taillight of the vehicle as a distinct object. These double detections tend to at least partially overlap. Therefore, a lower value for $\beta$ of 0.35 is used on dark scenes in order to reject the majority of the objects that overlap. The Fig.33 illustrates a TSI where the reflection of the headlights of one vehicle produced an additional undesired object, which was removed with the lower $\beta$.

![Fig.33: Left: $\beta = 0.35$; Right: $\beta = 0.65$](image)

### 3.8 Occlusion detection

As already explained in section 2.4.1, sometimes a vehicle view is obstructed by some other object. In order to tackle this problem, a solution using Multiple Virtual Detection Lines (MVDLs) was developed.

The proposed approach is partially based on the method presented in [Mithun, 2012], and explained in section 2.4.1.3. The method relies on drawing multiple virtual detection lines in different positions along the same lane and tries to find a correspondence between the vehicles identified in the
different resulting TSIs, checking for vehicles that may not be occluded in some of the TSIs to detect them.

The main difference between the proposed method used and the one presented by Mithun is the way the vehicles are associated between different TSIs.

Mithun’s method tries to associate the same vehicles between TSIs using their size and the order of the search. It associates the left most object of one TSI to the left most object of the following TSI and so forth. If the size dissimilarity between certain objects associated between TSIs is above a threshold, then it considers that there is occlusion. This strategy works well if vehicles don’t change their relative position between two consecutive VDLs and there is not too much noise in the video. Since in this method the objects between TSIs are simply associated by their distance to the beginning of the horizontal axis, if there is, for instance, an undesired object originated by noise detected as the leftmost object in $TSI_1$, but not detected in $TSI_2$, when the relationships are generated between objects in both TSIs, the noise of $TSI_1$ is linked to the leftmost vehicle of the $TSI_2$. Moreover, since the size of the object in $TSI_2$ is probably bigger than the noise detected as an object in $TSI_1$, the method will conclude that there is occlusion in $TSI_2$. Furthermore, all the remaining relationship generated between $TSI_1$ and $TSI_2$ are wrong, since the very first linked objects are incorrectly linked.

The method proposed here is based on the position of the vehicles observed in one TSI’s VDL and the expected position in the other TSI, which is estimated using the known positions of the VDL relative to each other. With this knowledge, it is possible to correctly associate the vehicles in both TSIs and signal possible undesired objects found throughout the TSIs, for instance due to noise. This avoids early erroneous occlusion detection of vehicles, assuring that the remaining objects, which do not have a correspondence to one of the TSIs, are either occluded vehicles or other moving objects detect in the TSI. The procedure for detection of objects which are possible occluded vehicles is summarized in the flowchart of Fig.34. The remaining possibly occluded objects found here are further analyzed in section 3.9.
Fig. 34: Flowchart of object occlusion detection

1. Generation of a second VDL

The implemented algorithm can automatically generate a second VDL for each VDL manually drawn by the user, in order to be able to perform the above occlusion detection analysis using MVDLs. Equations 3.18, 3.19 and 3.20 are used to acquire the angle between the first VDL with the x axis, to be able to generate the second VDL with the same orientation angle.

\[ l_1 = x_2 - x_1 \]  
\[ l_2 = y_2 - y_1 \]  
\[ \theta = \arctan \left( \frac{l_2}{l_1} \right) \]

where:
- \((x_1, y_1)\) and \((x_2, y_2)\) are the endpoints of the known VDL;
- \(l_1\) and \(l_2\) are the lengths between the two points in both axis;
- \(\theta\) is the angle of the VDL in respect to the x axis.

The endpoint 1 is always considered to be the one closest to the left side of the image and thus it satisfies \((x_1 \leq x_2)\). This condition is applied to restrict the result of \(\theta\) between the angle of \(-90^\circ\) and \(90^\circ\). This angle \(\theta\), is necessary to allow the generation of the second parallel VDL at distance \(d\) of the first VDL, as illustrated by Fig.35. In alternative, this second VDL can be manually defined by the user, if so desired.
2. Determination of first VDL crossed

In order to further improve the linkage of the objects between the generated TSIs, the direction the vehicles are moving across the VDL is taken into account. This allows a better estimate of the area where the vehicle’s position should be between TSIs. Knowing the direction allows to conclude that if the first VDL is crossed before then second VDL, then a vehicle detected in the first TSI in a certain position can never be placed to the left of this position in the second TSI, because it has a displacement to the right as illustrated in Fig. 36. This is due to the fact that the horizontal axis represents the time, advancing from left to right. To order the VDLs and their respective TSIs according to the time they are crossed by vehicles, both possibilities are tested, namely crossing $VDL_1$ and then $VDL_2$ or vice-versa. The sequence which presents a lower number of possible occlusions is considered the correct sequence. After acquiring the right sequence, the search for occluded objects is performed with the following conditions:

$$\begin{align*}
\begin{cases}
x_s = x_i + c, & \forall i \in A \\
y_s = y_i, & \forall i \in A
\end{cases}
\end{align*}$$

$$|x_s - x_j| < r_{max} \land |y_s - y_j| < r_{max}, \quad \forall j \in B$$

where:

- $A$ is the set of objects in $TSI_1$ and $B$ is the set of objects in $TSI_2$;
- $(x^i_1, y^i_1)$ is the center of an object $i$ in the first arbitrarily chosen $TSI_1$;
- $(x^j_2, y^j_2)$ is the center of an object $j$ in the other arbitrarily chosen $TSI_2$;
- $(x_s, y_s)$ be the estimated position of the object $i$ of $TSI_1$ in $TSI_2$;
- $r_{max}$ is the maximum distance from the estimated position to search for a object to associate;
- $c$ is a value in the interval $]0, \infty[$ added to the $x$ axis due to the displacement of the object between TSIs.
When 3.22 is satisfied the objects are linked, otherwise they are not linked together. The value \( r_{\text{max}} \) and \( c \) are considered directly proportional to the distance \( d \) assigned between VDLs following the equations 3.23 and 3.24:

\[
\begin{align*}
    r_{\text{max}} &= \left\lfloor ad \frac{\text{FPS}}{15} \right\rfloor \quad 3.23 \\
    c &= \left\lfloor d \frac{\text{FPS}}{15} \right\rfloor \quad 3.24
\end{align*}
\]

where:

- FPS is the value of frames per second of the video sequence;
- \( \alpha \) is a positive value chosen by the user

The default value of \( \alpha \) is set to 1. The assumption is that the vehicle position should not change significantly between both TSI and thus a radius proportional to the distance of the VDLs should be enough to link the same vehicle in both TSIs. The higher the distance between VDLs, the higher is the importance of the velocity in the position of the vehicles between TSIs. Hence, it is possible to adapt this \( \alpha \) value manually if the user sees it fit to improve the vehicle association. An example of the object search between TSIs is illustrated in Fig.36, where the red circumferences are the search areas of the objects on the second TSI with center \((x_s, y_s)\) and radius \( r_{\text{max}} \).

![Fig.36: Example of object association](image)

3. Search for possible occlusions

Objects which were signaled as possible occlusions are then verified according to their location on the TSI. Hence, a verification of the position of this possible occlusion is performed using the conditions of equation 3.25.

\[
    x_o < b_w \land x_o \geq TSI_{\text{rows}} - b_w \land y_o < b_h \lor y_o \geq TSI_{\text{columns}} - b_w \quad 3.25
\]

where:
O is the set of possible occluded objects;
- \((x_o, y_o)\) is be the center of an object \(o\) belonging to the set \(O\);
- \(TSI_{rows}\) is the number of rows of the TSI;
- \(TSI_{columns}\) is the number of columns of the TSI;
- \(b_w\) and \(b_h\) are the distance to the borders of the TSI used for object rejection.

This verification is used to reject objects which have their center near the border when condition 3.25 is satisfied. In case of the top and bottom borders, adjusted by parameter \(b_h\), the object may be the result of an overly stretched VDL which may detect the shadow or part of the object crossing a lane outside the scope of that VDL, as is the case illustrated in Fig.37. In case of the left and right borders, adjusted by the parameter \(b_w\), the object may be the result of a vehicle which is in the beginning or the end of the video sequence being analyzed and does not cross both VDLs.

The sequence of the VDLs according to the time they are crossed is finally decided by

\[
\begin{cases}
    occlusions_{12} \geq occlusions_{21}, & VDL_2 \text{ is crossed before } VDL_1 \\
    occlusions_{12} < occlusions_{21}, & VDL_1 \text{ is crossed before } VDL_2
\end{cases}
\]

3.26

The cases where the number of occlusions is the same, the orientation cannot be defined by this method and by default is assigned that the \(VDL_2\) is crossed before \(VDL_1\). These cases rarely happen and are usually a result of bad search parameters, which should thus be manually modified.

In Fig.37, it is possible to observe an occlusion example in the central TSI and the detected occluded vehicle in the right TSI with a white rectangle. On the central image there is also a black rectangle, corresponding to a detected object that due to its border position and the lack of correspondence to the right TSI, it is rejected as a candidate vehicle. It can also be seen in the right TSI that a small contour on the bottom edge of the image was ignored due to its small area as a result of the FVE techniques used.

![Fig.37: Left: Position of the lines in the video; Center: Green line contour detection and TSI with occlusion; Right: Red line contour detection and TSI with the detected occlusion.](image)

3.9 Vehicle Counting

The final step in the proposed system consists of computing the final number of vehicles which were crossing the selected lane for the duration of the video sequence, based on the information provided by the two TSIs analyzed thus far. To that effect, it is still necessary to decide from the possible occluded objects detected in the previous step, which ones are occluded vehicles
and are not simply undesired detected objects. Hence, a decision procedure has to be implemented as illustrated in Fig.38.

Fig.38: Flow chart of vehicle count acquisition

1. Selection of one TSI as reference

The implemented decision procedure uses the TSI of the VDL which is crossed first by the vehicles as the reference TSI and uses the information provided by the second TSI to perform the decisions. For simplicity, consider VDL1 as the first crossed line and VDL2 as the second crossed line.

2. Initial vehicle count

The initial vehicle count is basically computed using the initial number of objects found, excluding the rejected objects of the FVE and the possible occluded objects by the follow expression:

\[
v_{\text{count}} = o_{\text{total}} - (o_{\text{occluded}} + o_{\text{ignored}})
\]

where:
- \(v_{\text{count}}\) is the number of vehicles counted;
- \(o_{\text{total}}\) is the total number of objects;
- \(o_{\text{occluded}}\) are the possible occluded objects;
- \(o_{\text{ignored}}\) are the ignored objects that were on the border of the TSI.

3. Relaxed search to link possible occlusions

In order to avoid possible association of objects cut off by a value of the radius search, \(r_{\text{max}}\), too low in the previous step, causing undesired occlusion cases in both situations, a relaxed search to link only the possibly occluded objects is performed with a higher value of \(r_{\text{max}}\), which is, by default, 50% more than the value used in the previous search, although it can be manually adjusted. Any of
the objects linked in this second search are excluded from the remaining occluded objects and are added to the vehicle count.

The decisions which have to be performed to address the remaining possible occluded objects fall in one of the following situations:

- The object doesn’t have any object around the estimated position where it should be in the other TSI;
- The object has one or more objects with which it could associate in the other TSI. However, they are already linked to other objects.

The objects remaining in the first situation are rejected as they are considered noise in one of the TSIs, while the objects in the second situation are then accepted or rejected following condition 3.28:

\[ \text{area}_o (\alpha + 1) < \text{area}_i \] 3.28

where:

- \( \text{area}_o \) is the area of the possibly occluded object;
- \( \text{area}_i \) is the area of object which is inside the searched area;
- \( \alpha \) is a positive value.

When condition 3.28 is satisfied, the object is being linked with an object with an area at least \( \alpha \)% superior to their own area, then it is accepted as an occluded vehicle and the vehicle count is incremented, otherwise the object is not considered occluded by the object \( l \) found in the search. If there is any other object in the searched area, then it is also verified, otherwise the object \( o \) is rejected and not added to the vehicle count.

After finishing this process of decision to assess the possible occlusions detected, a final analysis of the bigger objects of both TSIs is executed to tackle undetected occlusions, which has to satisfy the following conditions in 3.29:

\[
\begin{cases}
(1 - \beta) \text{area}_a < \text{area}_b < (1 + \beta) \text{area}_a \\
\text{width}_a > (1 + \gamma) \text{width}_{\text{average}}_1 \\
\text{width}_b > (1 + \gamma) \text{width}_{\text{average}}_2 \\
\text{height}_a > (1 + \gamma) \text{height}_{\text{average}}_1 \\
\text{height}_b > (1 + \gamma) \text{height}_{\text{average}}_2
\end{cases}
\] 3.29

where:

- \( \text{area}_a \) and \( \text{area}_b \) are the area of the linked objects is both TSIs;
- \( \text{width}_a \) and \( \text{width}_b \) are the widths of the linked objects in both TSIs;
- \( \text{height}_a \) and \( \text{height}_b \) are the heights of the linked objects in both TSIs;
- \( \text{width}_{\text{average}}_1 \) are \( \text{width}_{\text{average}}_2 \) are the average widths of TSI1 and TSI2;
- \( \text{height}_{\text{average}}_1 \) are \( \text{height}_{\text{average}}_2 \) are the average heights of TSI1 and TSI2;
- \( \beta \) and \( \gamma \) are positive values.
The objective of verifying these conditions is to select objects which have large dimensions in both TSIs. These objects may contain occluded vehicles which were not found using the occlusion detection approach. The first condition ensures that the object in both TSIs must have a similar area and thus \( \beta \) has by default a small value of 0.2. The second and third conditions ensure that the widths of both objects are bigger than the average with and the fourth and fifth conditions ensure the same for the heights. The value \( \gamma \) is called comparison variable and is assigned to the value of 1. The assumption made is that if the width and the height of the objects have values superior to two times their averages, there is probably occlusion of some vehicles on that object.

An estimate of the number of actual vehicles occluded is performed by the following expressions 3.30 and 3.31.

\[
\begin{align*}
    r_1 &= \frac{\text{width}_a}{(1 + \epsilon)\text{width}_{\text{average}}_1} \\
    r_2 &= \frac{\text{height}_a}{(1 + \epsilon)\text{height}_{\text{average}}_1}
\end{align*}
\]

\( v_{\text{added}} = \min(r_1, r_2) \)

where: is a value in the interval

- \( \epsilon \) is a positive value;
- \( v_{\text{added}} \) is the number of the occluded vehicles’ estimate.

The value \( v_{\text{added}} \) is added to the vehicles counted and the final value for the vehicles counts is thus acquired. The value \( \epsilon \) is called the decision variable and adjusts the number of vehicles added. \( \epsilon \) has by default the same value of \( \gamma \), because if follows the same assumption that if \( \text{width}_a \) and \( \text{height}_a \) have a value superior to two times their averages, there is probably occlusion of some vehicles on that object. This estimate is quite conservative and only adds vehicles to the vehicle count in rare cases where almost undoubtedly there is an undetected occluded vehicle.

As a note, all the procedures used in the Occlusion Detection section 3.8 and Vehicle Counting section 3.9 could be implemented with more than two VDLs. However the complexity involved would increase proportionally, as all the objects have to be searched in all the other generated TSIs to link them, and the results would in the best cases barely improve as detailed in section 4.6.

3.10 Cumulative data treatment

The data acquired from feature extraction from the objects detected, namely their area, height and width, in multiple video sequences can be stored and used to improve the object validation on section 3.7 and vehicle counting on section 3.9, if the videos are provided by the same static video camera. Hence, this cumulative data of the features acquired throughout distinct video sequences,
when available, is used on the average values utilized across the whole system.

On each new video sequence analyzed, a selected set of values of each feature is added to the cumulative data of that featured and is followed by a computation of the cumulative moving average of each feature thus far. In most video sequences, only a portion of values of each feature is added to the cumulative moving average. The accepted values must be in a specified interval to be accepted. This interval is defined by the parameter $\alpha$. The parameter is used to reject a percentage of the feature value found in the current TSI. Hence, from all the feature values collected on the current TSI, the lower and higher $\alpha\%$ of these values is rejected and thus not included in the cumulative moving average.

The reason for this rejection is that in case of small values, they should be avoided because they may be extracted from objects which are just noise, or in case of big values, the objects may contain occluded vehicles. The default value for $\alpha$ is 0.20, which means that lower 20% and higher 20% of the values of each feature found in the new video sequence are not added to the cumulative moving average. When the number of features in a new TSI is lower than 4, this condition is not applied, because it would exclude most new values.

The cumulative moving data (CMA) of each feature is afterwards computed using the following expression:

$$CMA_n = \frac{x_1 + \ldots + x_n}{n}$$  \hspace{1cm} (3.32)

where:

- $n$ is the number of value added to the CMA;
- $x$ is the value of a certain feature in a certain object.

This cumulative data is used instead of only the data provided by the current video sequence when $n$ is above a specific value. By default, the cumulative data is used when $n$ is over 25. The assumption made is that over such a value of $n$, the features averages provided by the cumulative data can provide better results, when used on FVE, than only the feature averages provided by the feature values found on the current video being analyzed.

### 3.11 Detection Optimization

The detection of objects in any generated TSI needs to be adapted to different operational conditions during the day to obtain the best results. The main factor affecting all the LTSV cameras is the variation of the illumination level along the day. This causes the necessity of distinct approaches to the detection of vehicles depending on the available light on the scene observed by the camera. Therefore, a different approach is proposed and developed for the system using three modes:

- the Day Mode (DM);
- the Night Front Side Mode (NFSM);
- the Night Back Side Mode (NBSM).
Each of these modes has its own settings for vehicle detection and validation. The selection of the operation mode is illustrated in the flowchart of Fig.39. After knowing in which mode the scene is at the current moment, an optimization of the chosen sensitivity threshold value may be performed.

3.11.1 Mode Selection

Firstly, it is necessary to know if the scene is bright or dark. The scene is usually bright during the day and dark during the night, but it may not always be the case, especially if there are poor weather conditions. In case the scene is bright, there is no need to determine any more condition and bright detection settings of the DM are used on the TSI. However, if the scene is dark, it is necessary to know the direction in which the vehicles are moving. The main reason to assess the direction is related to the fact that in dark scenes, the vehicles have to be detected by the lights emitted by them and the headlights have a much higher intensity than the taillights, which cause reflection on the road. This reflection usually has a gap between the vehicle emitting it and its reflection on the floor, as illustrated in Fig.40, which may cause a double detection of the same vehicle. Knowing the direction, the settings of the edge detector can be optimized to just detect the headlights reflection or the taillights.

Fig.40: Double detection due to headlights' reflection

The scene is considered bright when condition 3.33 is satisfied. Otherwise the scene is
considered dark.

\[ I_{\text{average}} \geq \alpha \quad 3.33 \]

where:

- \( I_{\text{average}} \) is the average pixel intensity of the sample image provided by the video sequence of the camera;
- \( \alpha \) is a value defined between [0, 255].

The average intensity of the pixels provided by the sample image taken from the camera is computed and if it satisfies the condition 3.33, it is considered as a bright scene. Otherwise it is considered as a dark scene. The threshold has a default value of 95, which was acquired empirically, but it can be manually adapted. The assumption made here is that the night videos have a lower pixel average on their image than the day videos, which hold true to all the videos analyzed in which all night video had pixel average lower than 95, while day videos had pixels averages above this value. If it is a dark scene, the VDL sequence computed in the equation 3.26 in section 3.8 is used to know if the system has to detect the headlights on the NFSM or the body of the vehicles on the NBSM.

### 3.11.2 Settings of the modes

The proposed modes in this dissertation have distinct settings for the detection of objects and their validation as vehicles. Additionally, to the distinct settings, the two night modes perform some modification to the TSI to increase their vehicle counting accuracy. The proposed settings were developed to have a system more robust to variance of the illumination. Therefore, additionally to the three modes for day and night scenes, there are also adjustments on the detection parameters to the illumination changes which occur inside each of the modes.

The methods and parameters vary according to the mode to improve the object detection and vehicle validation. They are discussed in the following sections.

#### 3.11.2.1 Canny threshold

The canny threshold has to be correctly adjusted in order to have satisfactory vehicle detection and counting. This is especially important in the day mode (DM) and the Night Back Side Mode (NBSM). The approach developed has the goal of providing good vehicle detection without having to manually adjust this parameter each time for each TSI. Hence, a function which was based on the best detection results of the empirical experiments was developed.

This function relates the average intensity of the pixels in the analyzed TSI with the Canny threshold value which tended to obtain good object detections on the experimentally tested TSIs on that particular mode. Hence, a set of videos of each mode in the various operational condition tested were used, in order to generate the curve for each of the three modes. This operational conditions used on the tests are detailed in section 4.2. After acquiring empirically the values that obtained the best results on each video of each mode, a curve which best fitted the acquired data was generated.
for each of the modes. The obtained curves are presented and discussed along this section.

The curve of the function on the DM is illustrated in the following Fig. 41.

![Day Mode Curve](image)

**Fig. 41: Canny threshold curve on day mode**

The resulting curve shows that the threshold should increase exponentially with the average intensity of the pixels, which means the sensitivity of the Canny edge detector should decrease when the average intensity of the TSI’s pixels increases. The Canny threshold curve was set to not decrease below 200 to avoid too much sensitivity on the detector, which would increase the amount of noise detected as objects. On the other hand, the Canny threshold curve was set to not go above 700 to avoid becoming too insensitive to video sequences with high luminance.

The curves of the functions of the NBSM and NFSM are illustrated in the following Fig. 42.

![NBSM & NFSM Curve](image)

**Fig. 42: Canny threshold curve on night modes**

The resulting curve on the NBSM also increases with the average intensity of the pixels.
However, this time it increases logarithmically. This means that in this mode, even with higher pixel intensity averages, the sensitivity has to be higher than in the DM. The main cause is the lower contrast the night video sequences have compared to the day video sequences. Hence, the maximum value assigned to the Canny threshold is 625, which is lower than the maximum assigned to the day mode. A minimum of 275 was assigned to the value of the threshold, which is also slightly higher than the day mode minimum, due to the higher presence of noise in night video sequences which tend to be picked with lower thresholds.

The Canny Threshold does not vary on the NFSM. This is due to the Otsu’s binarization, which is applied to a modified TSI done with the value of the night saturation, which is explained in the section 3.11.2.2. Hence, the Canny edge detector is this mode is simply used to individualize the objects, since the contours are already defined by the binarization. Therefore, a constant value of 675 is applied to the Canny threshold as the detector does not need significant sensitivity on a binarized image.

3.11.2.2 Otsu’s binarization

For the NFSM a distinct approach was developed to detect the vehicles. While in DM and NBSM the vehicles are found by detecting the body structure of the vehicle on the TSI, on the NFSM the vehicles are found by detecting the reflection of the headlights of the vehicles on the pavement. This approach allows better detection of the vehicles due to the higher intensities of the pixels containing reflections of the headlights, which have a higher contrast with the background and thus are easier to detect.

In order to only acquire these reflections, an Otsu’s binarization is applied to a modified TSI, as illustrated in Fig.43, which is detailed in section 3.11.2.3. The Otsu’s binarization is a global thresholding technique, which reduces a gray level image to a binary image [Otsu, 1979]. This method assumes that the image can be separated into two classes. In this case, one class should be the headlights reflections and the other class all the rest. The method finds the best threshold separating both classes by minimizing the intra-class variance through an exhaustive search.

![Fig.43: Otsu’s binarization example](image)

3.11.2.3 TSI modifications

The TSI needs some modifications to improve the accuracy of the system when operating with dark scenes. These modifications are necessary to address the effect of vehicles’ lights, which tend to increase the vehicle counts due to double detections of the same vehicle (caused by light reflection in the pavement). These modifications are performed by a night parameter, which has a
different value and function in each of the two night modes. The performed modifications are explained in the next paragraphs.

On the NFSM, the goal is to detect the headlights of the vehicle. To perform this, the values below a certain threshold defined by the night parameter are modified on the TSI and assigned the threshold value, as shown on the following expression 3.34.

\[
\begin{align*}
I(x,y) &= I(x,y), & \text{if } I(x,y) \geq \alpha \\
I(x,y) &= \alpha, & \text{if } I(x,y) < \alpha
\end{align*}
\]

where:

- \(I(x,y)\) represents the TSI's pixel intensity in position \((x,y)\);
- \(\alpha\) is the night parameter.

The value assigned for \(\alpha\) is 150 for the NFSM. The assumption made here is that at night the pixel intensity of the vehicles tends to be below 150. Hence, assigning the night saturation to 150 will modify all the values of the TSI below this value to the threshold value of 150. The vehicles will not be highlighted by the Otsu's binarization and thus the vehicles structure will be ignored on the object detection, detecting instead its headlights reflection on the pavement.

On the NBSM, this night parameter is used with the opposite purpose. In this mode, the vehicles are detected by the structure of the vehicle or by the lights emitted by the taillights, depending on the angle of the camera with the road. However, due to the high intensity of the headlights, their reflection on the pavement might be also detected as separate moving objects, causing double detections of the same vehicle. Hence, the higher intensity pixels are truncated to avoid detecting the headlights of the vehicle following the condition 3.35.

\[
\begin{align*}
I(x,y) &= \alpha, & \text{if } I(x,y) \geq \alpha \\
I(x,y) &= I(x,y), & \text{if } I(x,y) < \alpha
\end{align*}
\]

where:

- \(I(x,y)\) represents the TSI's pixel intensity in position \((x,y)\);
- \(\alpha\) is the night parameter.

The value assign for \(\alpha\) in this case is 220. The idea followed here is that cutting off the highest pixel intensities of the TSI can avoid the detection of the headlights reflection on the pavement. As a result, the majorities of these reflections are ignored by the Canny detector and thus improve the vehicle counting accuracy on the NBSM.

### 3.11.3 Sensitivity threshold optimization

Once the type of mode present at that moment is acquired, the average intensity of the pixels generated by the TSIs are also computed to determine which sensitivity should be initially used on the edge detector. This initial sensitivity is defined using a function already generated empirically for each of the three types of mode, according to the value of the average intensity of the TSI pixels.
The initial sensitivity threshold value is based on the current state and the intensity of the pixels on the current TSI. To further improve the detection in the current conditions of the TSI, an optimization of this threshold used in performed. This optimization is done using the following condition 3.36:

$$\beta^* = \min_{\text{view}} (\text{area}_{\text{variance}}_i)$$  

The interval \([a, b]\) of the set \(X\) are defined by:

$$a = \beta - k$$  

$$b = \beta + k$$

where:

- \(X\) is a set of integer thresholds values;
- \(k \in \mathbb{Z}\) is the value which defines the size of the search interval;
- \(\text{area}_{\text{variance}}_i\) is the variance of the area of the objects in a certain TSI;
- \(\beta\) is the initial threshold value provided by the function of the defined state;
- \(\beta^* \in \mathbb{Z}\) is considered the optimal threshold in set \(X\).

A sweep across the interval \([a, b]\) of threshold values is performed and the value which has a lower area variance is chosen. It is basically a search for the lower local minimum on the chosen interval. The assumption made here is that most of the vehicles should have similar area, thus the threshold value with the minimum area variance probably provides the best detection in the actual video sequence provided by the camera’s scene with the current available luminosity. In the case where the chosen minimum is constant across an interval of threshold values of the set \(X\), the middle value of that interval is chosen as the sensitivity threshold to be used on the detection.
This chapter starts by summarizing and justifying the parameters used on the experiments, proceeds to detail the distinct operational conditions found in the experiments and subsequently compares the obtained results with works of other authors.
4.1 Overview

The objective of this chapter is to present the results of the performance of the system when using videos captured from live traffic cameras. This section provides details on the selection of the required system parameters and about the operational conditions of the videos used on the experiments. The accuracy and computational performance of the system are evaluated and the obtained results are compared with other published works.

The chapter starts with a discussion on the settings predefined for each system operation mode and the parameters used. It is followed, in section 4.2, by the specification of test conditions and videos used on the experiments. In section 4.3 the accuracy of vehicle counting is assessed in the various operation conditions discussed. Section 4.4 describes the computational performance of the developed system and 4.5 compares the results with other works addressing traffic management and vehicle counting.

4.2 Test Conditions

The videos used in the following experiments were acquired from LTSV cameras, mainly provided by Estradas de Portugal. These videos, as already stated before, have a resolution of 200x200, which is considered an extremely low resolution for video analysis. The video sequences provided are relatively short as they have a length of approximately 15 seconds, which was imposed by the available database. However, it is enough to assess the performance of the system. A list with information related to all the videos used in the experiments is provided in Annex 2 - Videos List.

The system was analyzed in distinct sets of operational conditions to allow a better understanding of its robustness to different situations. The next sections describe the type of operation conditions which were used in the different operational conditions that the system analyzed.

4.2.1 Videos with good illumination

This set of videos corresponds to the best operational conditions. The scene was captured with good illumination, which provides a good contrast between the road pavement and the moving vehicles travelling along.

These video sequences don’t present relevant problems, containing a limited amount of noise or involuntary movements of the camera, although some noise or jitter may be present in some cases. Sample images representative of this set of video sequences are included in Fig.44.
4.2.2 Videos with jitter

Video jitter consists in a random horizontal shift of each row of a video frame. It occurs when the synchronization row pulses are corrupted by noise or degradation of the storage medium or in wireless transmission [Nikolova, 2009].

The jitter in the video sequences causes the frames to be jagged near the objects which move between frames, as illustrated in Fig.45. When this happens over the VDL, it may cause double or more detections of the same vehicle or the detection of a vehicle with an extremely long width, which may be wrongly analyzed for possible occlusions as it will have a huge area compared to the other detected vehicles.

4.2.3 Videos with shadowed areas

These video sequences include areas of the road which are covered by shadows coming from buildings or trees, as illustrated in Fig.46. When a VDL is drawn over a shadowed area, it poses some difficulties on the selection of the best column for the background subtraction and also for the assignment of a good value for the Canny threshold.

These difficulties arise due to the contrast between the intensity of the pixels in the more illuminated and the shadowed areas. Hence, the system has to find a compromise allowing the detection of objects in both areas at the same time.
4.2.4 Videos with rain

The main issues with video sequences on raining weather are the water present on the pavement of the road which may cause undesired reflections and water drops which may be present on the camera lenses hindering the observation of passing vehicles.

The system has to adapt the detections on the TSI to tackle the extra noise present on these video sequences. Examples of rainy video sequences are illustrated in Fig.47.

4.2.5 Blurred videos

There are LTSV cameras that due to haze on their lenses or being out of focus generate blurred videos. This blur causes extra noise in the video which makes more difficult to individualize the moving objects, especially when vehicles move close to each other, causing more undesired vehicle occlusions.

The key to achieve good results in these situations is, if possible, to place the VDL in an area of the video where blur is less present or even non-existent. Some examples of blurred videos are illustrated in Fig.48.
4.2.6 Videos with moving camera

Video cameras used for traffic surveillance sometimes undergo a shaking movement, which can then be observed in the captured video sequences, affecting object detection and thus vehicle counting. This shaking is usually due to the presence of strong wind in the area where the camera is placed. This effect tends to reduce the efficiency of the background subtraction which becomes the main reason that causes an accuracy reduction in vehicle counting.

The videos used in this category present mild to moderate shaking as illustrated in Fig.49 for the video frames of a sample video sequence. When the shaking is too strong there is no way to acquire relevant information from the video sequence due to the resulting lack of video quality, not allow the estimation of a good background for the TSI and thus making any reliable vehicle counting impossible.

4.2.7 Dusk/Dawn video

The videos during dusk or dawn usually present some distinct characteristics from day videos with good illumination. The main differences are the lower average light intensity which causes lower contrast between the vehicles and the pavement and the presence of larger shadow areas created by the vehicles and surrounding structures, due to the lower angle of the Sun.

These videos require a higher sensitivity from the Canny edge detector and are characterized by a higher area of the moving objects found in the TSIs, due to an higher probability of detection of the vehicle shadow area connected with the vehicle’s body structure. Some examples of dusk videos are presented in Fig.50.
4.2.8 Night videos

The main propriety of night videos is their low luminance. This lower luminance makes it harder to distinguish the moving vehicles due to the lower contrast between them and the background. The existence of artificial light from the street lamps and the vehicles may cause dynamic shadows on the vehicles which change direction rapidly.

These shadows might cause occlusion of vehicles or might be considered a distinct vehicle, causing false positive detections. The reflection of the vehicle’s lights on the road pavement, especially the head lights, might additionally lead to double detections of the same vehicle, which raises the need to employ a special approach for dealing with vehicles moving towards the position of the camera, namely the NFSM operation mode.

The tests made when using night videos can fall into three distinct categories:

- videos with good illumination provided by surrounding street lamps;
- blurry videos caused by cameras slightly out of focus or in haze on the lenses;
- videos with jitter caused by faulty behavior of the LTSV system.

These three categories of videos are illustrated in Fig.51. On the left, there is an image of a video with good night illumination, on the center, a blurry video example and on the right an image showing jitter on one of the videos.

4.3 Vehicle Count Evaluation

The lack of a common benchmark database to compare different vehicle counting algorithms
makes harder the comparison of the obtained results [Buch, 2011]. Hence, the parameter used here to evaluate the performance of the implemented system is its vehicle counting accuracy. In addition, an efficiency analysis of the occlusion detection approach used on the developed system is included.

The principal causes that reduce the accuracy of the vehicles counts are:

- noise present on the video sequence due to the poor image quality provided by the LTSV cameras, causing false positive detections;
- operational conditions such as rain, bad lighting or darkness which impose lower contrasts between the roads and the moving vehicles and thus may lower the vehicle detected;
- LTSV imperfections due to video acquisition or transmission problems which cause jitter or blur to the frames of the video.

The accuracy was computed taking into account the number of vehicles crossing the observed road lane(s), the number of vehicles detected and undetected on the TSIs and the final vehicle count using the equation 4.1:

$$a_{cc} = \frac{v_{detected} + v_{undetected} + v_{false
detection}}{v_{detected} + v_{undetected} + v_{false
detection}} \times 100$$  \hspace{1cm} (4.1)

where:

- $a_{cc}$ is the accuracy in percentage;
- $v_{detected}$ number of vehicles correctly detected and counted on the analysis;
- $v_{undetected}$ number of vehicles not detected;
- $v_{false
detection}$ number of objects incorrectly counted as vehicles on the analysis.

The next paragraphs discuss the performance of the system modes on the distinct operations conditions explained on the previous section 4.2. It is followed by an evaluation of the performance when previous data about the features is known by the system.

### 4.3.1 Day mode (DM)

Table 1 summarizes the obtained vehicle count accuracies obtained by the developed system when operating in the day mode. The video sequences used consider variable traffic conditions, from low traffic/empty roads to high/congested traffic. This table provides information relative to the number of video sequences used and the number of vehicles present on each of the analyzed VDLs. The table also makes a comparison between the results obtained using the automatic parameters acquisition and a second approach where manual adjustment of parameters for each VDL was allowed to optimize results on each TSI. Each vehicle count is made separately with no previous data accumulated from previous training to learn the expected feature values for the objects crossing a specific VDL.
As expected, the accuracy is higher in video sequences with good illumination as the contrast between the vehicles and the background tends to be higher creating precise object detection as illustrated in Fig.52 in A. The video accuracy decreases in videos with jitter mainly due to double detections of the same vehicle caused by the jitter effect on the frames and subsequently on the TSI, as illustrated in Fig.52 in B, with the blue rectangles. The videos with shadowed areas present as main accuracy limitation the selection of the best Canny threshold values, to allow the detection of vehicles simultaneously on the shadowed and illuminated areas. This value has to be a tradeoff between the best values for detection in each of the areas. The desired performance cannot be always achieved, as illustrated in Fig.52 in C, where the vehicle in the dark area, highlighted with the red rectangle, was not detected.

The factor which lowered the accuracy in rainy videos was the presence of water drops on the lenses of the cameras, which may move during the video sequences causing dynamic noise and distortions on the generated TSIs, as illustrated in Fig.53 in blue in the left image. The blurred videos usually occur due to haze on the camera which is a similar situation to the rainy videos, as illustrated in Fig.53 on the right. However, this visual obstruction on the lenses does not move significantly during the video sequence and thus has a lower impact on the accuracy. Therefore, the main issue in both rainy video and blurred videos is the loss of contrast due to the lens obstruction, which may result in more undetected vehicle and undesired vehicle occlusions.

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>Automatic parameters</th>
<th>Manual parameters</th>
<th>Ground truth</th>
<th>Video sequences</th>
<th>Occlusions</th>
<th>Occlusions found</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good illumination</td>
<td>98.7%</td>
<td>100%</td>
<td>155</td>
<td>20</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>Jitter</td>
<td>94.6%</td>
<td>98.0%</td>
<td>146</td>
<td>20</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>Shadowed area</td>
<td>95.6%</td>
<td>96.7%</td>
<td>91</td>
<td>10</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Raining</td>
<td>91.8%</td>
<td>96.9%</td>
<td>98</td>
<td>11</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Blurred</td>
<td>94.2%</td>
<td>97.6%</td>
<td>206</td>
<td>20</td>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>Moving camera</td>
<td>90.0%</td>
<td>94.5%</td>
<td>129</td>
<td>17</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>Dusk</td>
<td>92.2%</td>
<td>96.1%</td>
<td>207</td>
<td>20</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>Averages/Totals:</td>
<td>93.8%</td>
<td>97.1%</td>
<td>1032</td>
<td>118</td>
<td>68</td>
<td>56</td>
</tr>
</tbody>
</table>

Table 1: Day mode accuracies
Fig. 53: Examples of TSIs in rainy and blurry videos

The videos captured with a moving camera present the worst accuracy, mainly due to the
difficulty to estimate the background of the TSI. When the camera shakes significantly, if the VDL
overlaps road markings, these markings tend to be detected as one or more vehicles as the
background subtraction cannot completely remove it as background. This situation is illustrated in the
two images of Fig. 54 by the objects detected by the red rectangles. Therefore, it is preferable to draw
the VDLs in areas with no road markings when cameras have a tendency to shake.

Fig. 54: Examples of markings detection in TSIs of shaking videos

The videos captured during dusk have a lower light intensity on the image and the vehicles
tend to present shadows due to a lower position of the sun relative to the vehicles. The issues
affecting the accuracy are:

- occlusions of vehicles due to a connection of the moving objects by their shadows;
- double detections of the same vehicle due to one object being its shadow and the
  other its body;
- vehicles which have their headlights already turned on, detecting the reflection of the
  light on the pavement as another object.

The obtained accuracy is thus mainly affected by the capacity to detect occlusions in the TSIs
and the ability to reject the double detections. The three issues are respectively illustrated in Fig. 55 by
the red highlighted rectangles in A, B and C.

Fig. 55: Examples of duck detection issues

The occlusions found when operating in day mode are caused mainly by shadows of vehicle
moving close to each other and by blur on the video which reduces the capacity of the system to
individualize the vehicles as single object. Hence, the videos captured at dusk and the blurred videos
tend to show more occlusions as confirmed by the tests reported on Table 1. The occlusion detection
by positioning of two parallel VDLs at 5 pixels of each other was able to find and correct 81.1% of the occlusions in the generated TSIs. This proves that using two VDLs for occlusion detection can be used as a tool to find the majority of the occluded vehicles. Fig.56 illustrates a detected occlusion due to the separation of the big object, in the first TSI, into two objects, in the second TSI. Fig.57 illustrates in red rectangles an occlusion which was not detected due to the proximity kept by both vehicles in the two TSIs.

![Fig.56: Example of detected occlusion](image)

![Fig.57: Example of undetected occlusion](image)

### 4.3.2 Night modes (NBSM & NFSM)

Table 2 summarizes the obtained vehicle counting results for the proposed system when operating in one of the considered night modes: Night Front Side Mode (NFSM) or Night Back Side Mode (NBSM). The presentation of results follows the same structure of the previous section. Therefore, each count is made separately with no previous information about the expected object features in each VDL.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Manual parameters</th>
<th>Ground truth</th>
<th>Video samples</th>
<th>Occlusions</th>
<th>Occlusions found</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NBSM - Good Illumination</strong></td>
<td>93.6%</td>
<td>97.3%</td>
<td>111</td>
<td>20</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td><strong>NBSM – Blurred</strong></td>
<td>90.6%</td>
<td>95.3%</td>
<td>128</td>
<td>20</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td><strong>NBSM – Jitter</strong></td>
<td>90.2%</td>
<td>94.4%</td>
<td>143</td>
<td>20</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td><strong>NFSM - Good lighting</strong></td>
<td>89.4%</td>
<td>92.1%</td>
<td>114</td>
<td>20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>NFSM – Blurred</strong></td>
<td>90.6%</td>
<td>92.8%</td>
<td>180</td>
<td>20</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td><strong>NFSM – Jitter</strong></td>
<td>89.0%</td>
<td>91.9%</td>
<td>135</td>
<td>19</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Averages/Totals</strong></td>
<td>90.6%</td>
<td>94.0%</td>
<td>811</td>
<td>119</td>
<td>32</td>
<td>23</td>
</tr>
</tbody>
</table>

Table 2: NBSM and NFSM accuracies

The obtained accuracies on these two modes are slightly inferior to those obtained when
operating in day mode. This is expected due to the lower luminance levels and resulting lower contrast found in the generated TSIs.

It is interesting to note the low variance in the observed accuracies in most of cases, especially in the NFSM mode, where the Blurred videos have actually a slightly better accuracy than the videos captured with good lighting. This shows that the use of an Otsu binarization in the NFSM, in order to just detect the headlights reflection on the pavement, made this operation mode more robust to noise. However, this is also the operation mode with lower accuracies, mainly due to a higher number of false positive detections of vehicles. The false positives detections are typically caused by multiple detections of the same vehicles, as illustrated in Fig.58 by the small red rectangles.

![Fig.58: Examples of NFSM double detections](image)

The same approach with VDLs distancing 5 pixels was used on the night operation modes to detect occlusions. On the night modes, the percentage of detected occluded vehicles decreased slightly, to 71.9%, due to the lower capacity to differentiate moving objects at night caused by the lower luminance. The number of occlusions on the NFSM is almost null due to the different approach using the Otsu’s binarization to just find the reflection of the head lights of the vehicles on the road pavement.

### 4.3.3 Cumulative data treatment results

This section reports the performance evaluation computed when considering a set of 16 video sequences, all captured by the same video camera, pointing at the same road lane without any position or angle variation. These video sequences include 12 videos with good illumination and 4 videos in dusk. They are used to compare the results of vehicle counting using the cumulative learned information of features as described in section 3.10 versus the results of analyzing each video independently, i.e., without learning any previous information about vehicle features.

Two sets of performance tests were conducted to evaluate the effect of cumulative data treatment. In both cases two VDLs were placed in the exact same coordinates.

For the first set of experiments, 12 videos with good illumination were used. This set of experiments used 11 of the 12 videos for training and used the remaining video to evaluate the results. The process was repeated 12 times in order to test all the possible combinations.

For the second set of experiments, 16 videos were used. The 12 videos with good illumination from the previous set and 4 additional dusk videos. This set of experiments used 15 of the 16 videos for training and used the remaining video to evaluate the results. The process was repeated 16 times in order to test all the possible combinations. The goal is to verify if using videos with different operational conditions in the cumulative data can improve the overall accuracy of the system.
All the system parameters were set automatically in both methods. The results of both experiments are presented on Table 3.

<table>
<thead>
<tr>
<th>Sets</th>
<th>Accuracy with untrained data</th>
<th>Accuracy with trained data</th>
<th>Total number of vehicles counted on test videos</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.5%</td>
<td>96.9%</td>
<td>91</td>
</tr>
<tr>
<td>2</td>
<td>94.7%</td>
<td>97.2%</td>
<td>127</td>
</tr>
</tbody>
</table>

Table 3: Accuracy with trained and untrained data

The results of the first set of experiments show that the accuracy with trained data tends to slightly improve the vehicle counting accuracy. The results of the second set of experiments also show accuracy improvements compared to the accuracy with untrained data. As a result, we can conclude that the use of cumulative data treatment tends to improve the quality of the results that can later be achieved when analyzing videos, even if captured in different conditions.

4.4 Computational Performance

The principal element which affects the time and speed of the various steps of the developed system is the spatial and temporal resolutions, and also the duration of the video sequence. This is due to the increase of computational resources necessary as a result of bigger TSIs which need to be analyzed.

The tests reported on this dissertation were run on a personal desktop computer equipped with an Intel Core i7 CPU, 4 GB of RAM on a 64-bit Ubuntu 12.04 operating system, which can be considered moderate computational resources. The running times presented in Table 4 were obtained considering 2 VDLs, with the second VDL being generated automatically based on the position of the first one. It includes the whole processing time since the first VDL is selected by the user until the final vehicle count is displayed. The tests were made on low resolution and high resolution video sequences of 15 seconds duration, with 15 FPS for the low resolution videos sequences, and 19 seconds with 30 FPS for the high resolution ones. The set of good illumination videos was used as test set for the low resolution videos, with one run on each video, while a single high resolution video, showing several roads was used for all the runs, considering different positioning of the VDL.

<table>
<thead>
<tr>
<th>Video Resolution</th>
<th>Average running time</th>
<th>Average time per frame</th>
<th>Number of runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>200x200</td>
<td>0.357 s</td>
<td>1.47 ms</td>
<td>20</td>
</tr>
<tr>
<td>1920x1080</td>
<td>10.39 s</td>
<td>21.64 ms</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 4: Computational performance

As expected, the running time increases with the resolution of the video. The main reason is the bigger size of the images on the high resolution video sequence. The higher resolution images need more resources from the system to process and to generate the TSIs. The TSIs also tend to have larger dimensions, because there are more pixels detailing each road lane crossed by the VDLs. The high resolution video used has a higher FPS rate than low resolution videos, which also increases
the running time. However, the running times are still significantly lower than the duration of the videos being analyzed. On the low resolution videos, they are around 42 times lower and on the high resolution videos, they are around 2 times lower. This still allows the system to be used on real-time situations in both cases.

4.5 Results comparison

This section provides a comparison of the performance results of the developed system with other developed works. The literature related with vehicle counting tends to focus on their performance tests on videos with good lighting [Buch, 2011]. Hence, the good lighting accuracy of the developed system is used as comparison of the results on the Table 5 in this section.

The comparisons were made with five other systems. In [Rivlin, 2002], a background-subtraction-based (BSB) method with a heuristically chosen threshold is used to individualize and count the vehicles. The systems developed by [Hue, 2009], [Rashid, 2010] and [Santos, 2011] use a single VDL (SVDL) approach for detection of the vehicles, while [Mithun, 2012] and the system developed in this dissertation use a multiple VDL (MVDL) approach.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>Vehicles counted</th>
<th>Ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSB [Rivlin, 2002]</td>
<td>86.2%</td>
<td>306</td>
<td>355</td>
</tr>
<tr>
<td>SVDL [Hue, 2009]</td>
<td>92.7%</td>
<td>329</td>
<td>355</td>
</tr>
<tr>
<td>SVDL [Rashid, 2010]</td>
<td>92.7%</td>
<td>329</td>
<td>355</td>
</tr>
<tr>
<td>SVDL [Santos, 2011]</td>
<td>96.7%</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MVDL [Mithun, 2012]</td>
<td>98.3%</td>
<td>349</td>
<td>355</td>
</tr>
<tr>
<td>Proposed system</td>
<td>98.6%</td>
<td>292</td>
<td>296</td>
</tr>
</tbody>
</table>

Table 5: Results comparison

The obtained accuracy of the developed system is clearly superior to the method using a BSB approach and has a significant accuracy improvement compared to the SVDL approach. The system has slightly superior accuracy compared to the MVDL system developed by [Mithun, 2012]. It would be interesting to compare the performance of both systems on night videos, however there were no tests performed on such operational conditions by the authors of the paper.

4.6 Performance with 3 or more VDLs

This section details the effect on the performance, when using more VDLs for vehicle counting. The tests were made considering several operational conditions and for varying amounts of traffic on the analyzed road lanes. The procedure applied to analyze results when using more TSIs for the same road lane was simply a repetition of the process applied to 2 TSIs. The result of the analysis of the first 2 TSIs is used to compare with the third TSI. Then that result can be used to compare with a fourth
TSI, and so on.

The results of the experiments are present in Table 6, divided according to the traffic present on the video. Videos with less than 10 vehicles in 15 seconds were considered low to medium traffic, while videos with ten or more vehicle were considered with high traffic.

<table>
<thead>
<tr>
<th>Traffic</th>
<th>2 VDLs</th>
<th>3 VDLs</th>
<th>4 VDLs</th>
<th>Ground Truth</th>
<th>Number of videos</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low/Medium</td>
<td>96.1%</td>
<td>95.2%</td>
<td>94.0%</td>
<td>103</td>
<td>20</td>
</tr>
<tr>
<td>High</td>
<td>94.6%</td>
<td>96.4%</td>
<td>94.1%</td>
<td>264</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 6: Accuracies for 2, 3 and 4 VDLs

On one hand, the obtained results on videos with low to medium traffic show that increasing the number of VDLs usually doesn’t improve the accuracy of the vehicle counting. The accuracy actually tends to decrease as the number of VDLs increases. The reason is that sometimes the system fails to link correctly the objects between TSIs and thus they are wrongly rejected. This is illustrated in Fig. 59, for the object in red on the third TSI, which was not linked to the corresponding vehicle of VDL2 due to the distance difference between their locations on the TSIs.

On the other hand, the results for high traffic when increasing the number of VDLs tend to improve. This is due to the rejection of objects caused by noise or jitter, which were present on the first 2 TSIs, but were not present on the third or the fourth TSIs. Hence, when the system detects a high flux of vehicles, the number of VDLs used should increase to 3 for better results. However, the use of 4 VDLs reduces the accuracy of the system. The reason is the same as in the low to medium traffic experiments. The development of an improved linkage method between objects on the distinct TSIs would allow the use of more VDLs to increase the overall accuracy of the system.
This chapter finalises this work, summarising conclusions and pointing out the system strengths and weaknesses and also indicating aspects to be developed in future work.
5.1 Overview

The developed Video-based Traffic Management system achieves its goals to be efficient and flexible method to count vehicles. It can work without any necessary adjustment by its operator, independently of the operational conditions it encounters. The system can be used as an efficient and low cost replacement for other vehicle counting methods described in section 2.2.

The continuous improvements on the computer vision field of investigation can turn Video-based Traffic Management has a primary option to analyze traffic information. This system has the advantage of being able to utilize already implemented LTSV systems and facilitate the analysis of the road traffic by traffic operators, while avoiding the high installation and maintenance cost of alternative systems for vehicle counting such as induction loops.

The following sections describe the strengths and weaknesses of the developed system and suggest future improvements to increase the performance on the system.

5.2 Strengths

The use of more than one VDL to analyze the features extracted from the TSIs, allows the development of a more flexible, accurate and robust Video-based Traffic Management System as described in the following sections.

5.2.1 Flexibility

The developed system has a high degree of flexibility offered to its operator summarized in the following three points:

- there are no limitations to the number of VDLs used on each video nor angle and length limits to them;
- there are no limits for the resolutions of the videos, their duration or their resolution;
- all the parameters used on the developed system can be adapted by the operator.

5.2.2 Easy-to-use

The empirically acquired parameters assigned to the various functions used on the system allow even an untrained operator to use the software, since these parameters are automatically adjusted accordingly to the situation. The operator just has to select the road lanes which are going to be analyzed and the software will perform the rest of the computations without any necessary interference from the user and give the final vehicle count.

5.2.3 Performance & Efficiency

The system has a good performance and robustness in all the operational conditions, especially on DM video sequences, where it can achieve an accuracy of almost 99% for automatic
parameters and 100% for manually adjusted parameters. Even on dark videos the accuracy rounds the 90% with automatics parameters and 94% with manually adjusted parameters.

The vehicle count is acquired with a limited used of computational resource for videos of any size and duration. The processing time rarely surpasses 0.5 seconds for low resolution video sequences of 15 seconds, which allow this system to be also used in real time situations.

5.3 Weaknesses

The performance and efficiency of the developed Video-based Traffic Management System is dependent of the quality of the TSIs it can generate. Hence, the factors which affect the quality of the TSIs are the main weaknesses found in this vehicle counting approach, which are discussed in the following paragraphs.

5.3.1 Camera Movements

The continuous shaking movements of the camera cause distortions on the TSI, which reduce the efficiency of the background estimation and subtraction. This situation might even turn the vehicle count impossible if the shaking is extremely accentuated. An image stabilization algorithm could be used to improve the video quality for detection.

5.3.2 Camera Position & Angle

The cameras should be positioned as high and as vertical as possible to the road to obtain the best performance. The Fig.60 shows distinct examples of cameras positions and angles to the road. If the camera is too close to the ground, as the middle image, or has an angle almost parallel to the road pavement, as the left image, the system parameters have to be adjusted manually, since the camera doesn’t have a good angle or position. On the other hand, the right image shows an example of a camera well positioned and with a good angle, which tend to obtain the best results as the probably of occurrence of vehicle occlusions is reduced.

Fig.60: Examples of cameras with different heights and angles

5.3.3 Light reflections & shadows

The light reflections at night represent a problem in the vehicle detection due to possible double detections by the head lights of the vehicles. The shadows which can be created by the sun,
street lamps or the vehicles’ lights might also cause multiple detections of the same vehicle or occlusion of closely moving vehicles. Those two situations are illustrated on Fig. 61, where on the image on the left the reflection of the headlights of the vehicles can be clearly seen and on the image on the right the shadows of the vehicles are overlapping other vehicles and may be detected as one object.

![Fig. 61: Examples of light reflections and shadows](image)

### 5.4 Future Work

There are some aspects and features which could be used to improve on the system with further investigation on the use TSIs for vehicle detection and counting. This topic is discussed in the following paragraphs.

The TSIs approach presented here for detection does not take any advantage of the color channels which may be available on the video. In this dissertation, the TSIs are always converted to a grayscale format for analysis. This color could probably be used, for instance, to detect the red taillights of the vehicles at night and use it as an alternative way to count vehicle in scenario with high tendency for the occurrence of occlusions. This could be especially helpful on occlusions caused by shadows, because the presence of shadows tends to not interfere significantly with the lights emitted by the vehicles.

The width of the objects on the TSI varies accordingly to the speed the object crosses the VDL and also accordingly to the FPS rate of the camera. This width could be used to have an estimate of the velocity of the vehicle. To work correctly, the software needs to have enough long term information of the objects’ features with the camera on a constant angle and position. This could be used to know if the traffic velocity is changing and could help predict possible traffic congestions before it occurs.

This Video-based system could also be integrated with another sensing system, such as inductive loops allowing the verification of possible malfunctions on the inductive loop count. This would provide as an alternative to the tedious job that an operator may have of verifying if the inductive loops are functionally correctly counting the vehicle crossing it. The use of a VDL line on a camera pointing at the location of the inductive loop could do it instead of the human operator.
Annex 1

Graphical User Interface

Details how the developed system performs when using a higher number of VDLs on each road lane analysis.
Graphical User Interface

The user graphical interface was created using the graphical design tools provided by Qt Creator Integrated Development Environment (ISE). The graphical interface is illustrated in Fig.62, Fig.63 and Fig.64.

The user should start by choosing the video which is going to be analyzed. This task is performed writing the address of the video in the area highlighted in rectangle number 1 and subsequently loading it. The video may also be browsed pressing the ‘…’ button. Optionally, the user may wish to play the video, which can be done pressing the button ‘Play’.

After loading, a sample image of the video sequence is displayed in rectangle 3. Some properties, such as the resolution of the video, are also shown here. The direction of the vehicles depends naturally of the VDLs being analyzed at the moment. This sample image can be used to select the VDLs to be generated. Alternatively, the coordinates of the endpoints of the VDL can also be specified in rectangle 3 and the line generated by pressing the draw button.

The generated VDLs are added to the list displayed in rectangle 2. This list shows all the generated VDLs, which are always added to the bottom of that list once generated. The position of the VDLs on the list can be modified using the buttons on the right of the list. The VDLs can also be removed from anywhere on the list in these buttons. The update button is used to update all the information related to the analysis of the video sequence displayed on the program. The two VDLs on top of the list are always the ones which are been shown on rectangles 4 and 5.

The rectangle 4 and 5 display the analysis made to the TSIs generated by VDLs selected from the list. The rectangle 4 displays the VDL which is crossed first by the vehicles. The rectangle 5 the VDL crossed afterwards. This is independent of the position of the top two lines on the list. The first image on each of these rectangles is the result of the contours of the canny edge detector on the TSI. The second image is the result of the feature extraction. It highlights the rectangles around the objects found and its centroids. The last image shows the result of the association of the objects between TSIs. Objects with white rectangles indicate possible occlusions and black rectangles indicate ignored objects. All the remaining colors are used to see the match done between the vehicles between the TSIs. Below the images, the number of objects found in each TSI is shown. The final vehicle count, obtained after the procedures made in the steps described in the previous sections is shown in rectangle 7.

Rectangle 8 displays the result of the commands send to the program by the user and may warn the user of possible errors in the options chosen for the analysis. The rectangle 9 shows the information related to the cumulative data stored about the averages of the objects detected on the first VDL across all the video sequences in which that VDL was present thus far.

Fig.63 shows the settings being used by the Canny detector on the current analysis of the displayed TSIs. These values can be manually adjusted by the user. Fig.64 has all the parameters being used on the system at the moment. These variables can be modified by the user. It provides the
option to change the distance used on the automatic generation of the second VDL and also give the option to manually generate the second VDL. It also includes the information related to the threshold being used for headlights detection at the moment if the scene is dark, which is not the case of this video sequence. The optimization settings are also included on this window.

Fig. 62: Main window
Fig. 63: Canny settings window

Fig. 64: Parameters window
Annex 2

Videos List

Complete list of the videos used on the experiments, including its location and length.
<table>
<thead>
<tr>
<th>Operational Conditions</th>
<th>Name</th>
<th>Location</th>
<th>Duration (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good lighting</td>
<td>Good1</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good2</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good3</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good4</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good5</td>
<td>Alto Barra</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good6</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good7</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good8</td>
<td>Alto Barra</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good9</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good10</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good11</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good12</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good13</td>
<td>Cruz Quebrada</td>
<td>14</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good14</td>
<td>Cruz Quebrada</td>
<td>14</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good15</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good16</td>
<td>A33</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good17</td>
<td>Encarnação</td>
<td>12</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good18</td>
<td>Alfragide</td>
<td>14</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good19</td>
<td>Cruz Quebrada</td>
<td>14</td>
</tr>
<tr>
<td>Good lighting</td>
<td>Good20</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter1</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter2</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter3</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter4</td>
<td>A43</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter5</td>
<td>Caxias</td>
<td>15</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter6</td>
<td>Cacém</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter7</td>
<td>Rio de Mouro</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter8</td>
<td>Alto do Forde</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter9</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter10</td>
<td>Av. de Ceuta</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter11</td>
<td>Laranjeiras</td>
<td>17</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter12</td>
<td>Alto do Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter13</td>
<td>A33</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter14</td>
<td>IC20</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter15</td>
<td>Alto de camarate</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter16</td>
<td>Cacém</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter17</td>
<td>Alto de camarate</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter18</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter19</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>Jitter20</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy1</td>
<td>Av. de Ceuta</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy2</td>
<td>TúnelGrilo</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy3</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy4</td>
<td>Alto do Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy5</td>
<td>IP7</td>
<td>16</td>
</tr>
<tr>
<td>Rain</td>
<td>Rainy6</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk1</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk2</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk3</td>
<td>Paredé</td>
<td>15</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk4</td>
<td>Alfragide</td>
<td>15</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk5</td>
<td>Belas</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk6</td>
<td>Pontinha</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk7</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>----------</td>
<td>----</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk8</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk9</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk10</td>
<td>Paíões</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk11</td>
<td>IC19</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk12</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk13</td>
<td>Bobadela</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk14</td>
<td>Bobadela</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk15</td>
<td>IC2</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk16</td>
<td>IP7</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk17</td>
<td>Av. de Ceutra</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk18</td>
<td>Laranjeiras</td>
<td>16</td>
</tr>
<tr>
<td>Dusk</td>
<td>Dusk19</td>
<td>Alto de Camarate</td>
<td>16</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area1</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area2</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area3</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area4</td>
<td>Paço de Arcos</td>
<td>9</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area5</td>
<td>Paço de Arcos</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area6</td>
<td>Alto da Barra</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area7</td>
<td>Alto da Barra</td>
<td>15</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area8</td>
<td>Laranjeiras</td>
<td>16</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area9</td>
<td>Laranjeiras</td>
<td>16</td>
</tr>
<tr>
<td>Shadow Area</td>
<td>Shadow_Area10</td>
<td>Alto Lumiari</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry1</td>
<td>Ameixoeira</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry2</td>
<td>Camarate</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry3</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry4</td>
<td>IC2</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry5</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry6</td>
<td>IP7</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry7</td>
<td>Ameixoeira</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry8</td>
<td>A33</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry9</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry10</td>
<td>Av. de Ceuta</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry11</td>
<td>Cril</td>
<td>17</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry12</td>
<td>Alto do Forte</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry13</td>
<td>Palácio Queluz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry14</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry15</td>
<td>IC17</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry16</td>
<td>Alto do Forte</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry17</td>
<td>IP4</td>
<td>15</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry18</td>
<td>IP4</td>
<td>15</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry19</td>
<td>Portela</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>Blurry20</td>
<td>IP7</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 7: List of DM videos
### NBSM Videos

<table>
<thead>
<tr>
<th>Operational Conditions</th>
<th>Name</th>
<th>Location</th>
<th>Duration (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good lighting</td>
<td>NB_Good1</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good2</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good3</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good4</td>
<td>Crl</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good5</td>
<td>Alto Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good6</td>
<td>Alto Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good7</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good8</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good9</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good10</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good11</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good12</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good13</td>
<td>A28</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good14</td>
<td>A43</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good15</td>
<td>Odivelas</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good16</td>
<td>Laranjeiras</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good17</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good18</td>
<td>Ameixeira</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good19</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NB_Good20</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter1</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter2</td>
<td>IP7</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter3</td>
<td>Crl</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter4</td>
<td>Alto do Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter5</td>
<td>Odivelas</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter6</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter7</td>
<td>A28</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter8</td>
<td>Mirafores</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter9</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter10</td>
<td>Alfragide</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter11</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter12</td>
<td>Paíes</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter13</td>
<td>Rio de Mouro</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter14</td>
<td>Laranjeiras</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter15</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter16</td>
<td>Ameixeira</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter17</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter18</td>
<td>Alto do Duques</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter19</td>
<td>IC17</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NB_Jitter20</td>
<td>Pontinha</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry1</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry2</td>
<td>Av. Padre Cruz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry3</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry4</td>
<td>A43</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry5</td>
<td>Odivelas</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry6</td>
<td>Paíes</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry7</td>
<td>Praça José Queirós</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry8</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry9</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry10</td>
<td>Carcavelos</td>
<td>15</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry11</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry12</td>
<td>Algés</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry13</td>
<td>Pontinha</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry14</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
</tbody>
</table>
Table 8: List of NBSM videos

<table>
<thead>
<tr>
<th>Operational Conditions</th>
<th>Name</th>
<th>Location</th>
<th>Duration (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good lighting</td>
<td>NF_Good1</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good2</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good3</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good4</td>
<td>Alto Barra</td>
<td>9</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good5</td>
<td>Alto Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good6</td>
<td>IP7</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good7</td>
<td>IC17</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good8</td>
<td>IC19</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good9</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good10</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good11</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good12</td>
<td>Alto Barra</td>
<td>15</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good13</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good14</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good15</td>
<td>A20</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good16</td>
<td>A28</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good17</td>
<td>A28</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good18</td>
<td>A43</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good19</td>
<td>A43</td>
<td>16</td>
</tr>
<tr>
<td>Good lighting</td>
<td>NF_Good20</td>
<td>Alto dos Duques</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter1</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter2</td>
<td>Alto do Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter3</td>
<td>Odivelas</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter4</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter5</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter6</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter7</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter8</td>
<td>A1</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter9</td>
<td>Miraflores</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter10</td>
<td>Pontinha</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter11</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter12</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter13</td>
<td>Piaões</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter14</td>
<td>Rio de Mouro</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter15</td>
<td>Benfica</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter16</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter17</td>
<td>2ª Circular</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter18</td>
<td>Ameixeoeira</td>
<td>16</td>
</tr>
<tr>
<td>Jitter</td>
<td>NF_Jitter19</td>
<td>Alto Lumiar</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry1</td>
<td>IC19</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry2</td>
<td>IC2</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry3</td>
<td>Calçada de Carriche</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry4</td>
<td>Cacém</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry5</td>
<td>Odivelas</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NB_Blurry6</td>
<td>Av. Padre Cruz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry7</td>
<td>Carcavelos</td>
<td>15</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
<td>------------</td>
<td>----</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry8</td>
<td>PalácioQueluz</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry9</td>
<td>Algés</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry10</td>
<td>Pontinha</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry11</td>
<td>Algívelas</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry12</td>
<td>Pina Manique</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry13</td>
<td>Alfragide</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry14</td>
<td>QuatroCaminhos</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry15</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry16</td>
<td>Tercena</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry17</td>
<td>Paiões</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry18</td>
<td>Rio de Mouro</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry19</td>
<td>Rio de Mouro</td>
<td>16</td>
</tr>
<tr>
<td>Blurred</td>
<td>NF_Blurry20</td>
<td>Praça José Queirós</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 9: List of NFSM videos
References


[FLIR, 2014t] Information about FLIR: http://flir.com/aboutFLIR/


