

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL  
INSTITUTO DE INFORMÁTICA  
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

VITOR DE OLIVEIRA GOMES

**A Novel Image-Based Approach for  
Detecting Vehicles in User-defined Virtual  
Inductive Loops**

Thesis presented in partial fulfillment  
of the requirements for the degree of  
Master of Computer Science

Advisor: Prof. Dr. Jacob Scharcanski

Porto Alegre,  
2015

## CIP – CATALOGING-IN-PUBLICATION

Gomes, Vitor de Oliveira

A Novel Image-Based Approach for Detecting Vehicles in User-defined Virtual Inductive Loops / Vitor de Oliveira Gomes. – Porto Alegre: PPGC da UFRGS, 2015.

77 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2015. Advisor: Jacob Scharcanski.

1. Vehicle detection. 2. Shadow detection. 3. Traffic monitoring systems. 4. Virtual loops. 5. Image-based sensors. I. Scharcanski, Jacob. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos Alexandre Netto

Vice-Reitor: Prof. Rui Vicente Oppermann

Pró-Reitor de Pós-Graduação: Prof. Vladimir Pinheiro do Nascimento

Diretor do Instituto de Informática: Prof. Luis da Cunha Lamb

Coordenador do PPGC: Prof. Luigi Carro

Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

*“Be wary of folly.”*

— ABENTHY

## ABSTRACT

The number of vehicles circulating in modern urban centers is increasing and traffic data have important applications for urban planning and development. Therefore, traffic monitoring systems are essential for managing modern urban centers and roadways. Nowadays, due to the advances in computer vision, there is an ongoing research effort to replace traditional traffic monitoring sensors (e.g. inductive loops, microwave, etc.) by image-based systems, which are easier to repair and are more flexible to operate. These image-based systems have to be robust against challenging situations that can severely affect the results, such as sudden illumination changes, cast shadows and partial or total occlusions. In this work, our main contributions are twofold: First, we propose a novel shadow detection scheme that combines hypergraph image segmentation with color and texture information for accurate shadow detection. Second, a vehicle detection scheme that uses an image-based sensor (camera) for detecting vehicles at user-defined virtual loops, simulating the operation of inductive loops. Using the proposed vehicle detection scheme, allows several user-defined virtual loops to be monitored simultaneously with one camera. Initially, the background is modeled with an improved Gaussian Mixture Models approach, and then the shadows detected at the virtual loops are removed to minimize false vehicular detections. Next, vehicles are detected at the user-defined virtual loops using a combination of efficient edge detection and color information. The proposed vehicle detection scheme provides a vehicle detection signal that also can be used for counting vehicles at the user defined virtual loops. The experimental results indicate that the proposed scheme can accurately detect vehicles at user-defined virtual loops and provide their counts (with more than 98% accuracy, in average), besides being more robust to cast shadows and sudden illumination changes than comparable methods that represent the state of the art. Also, experiments performed on common and publicly shadow detection datasets suggest that our proposed hypergraph image segmentation shadow detection approach is capable of obtaining accurate shadow detection results when compared with recent and relevant shadow detection methods.

**Keywords:** Vehicle detection. Shadow detection. Traffic monitoring systems. Virtual loops. Image-based sensors.

## RESUMO

O número de veículos em circulação nos grandes centros urbanos está aumentando e dados de tráfego tem importantes aplicações para o planejamento e desenvolvimento urbano. Portanto, esses dados são essenciais para o gerenciamento de cidades e rodovias. Hoje, devido aos avanços em visão computacional, há diversas pesquisas focadas em substituir sensores de monitoramento de tráfego tradicionais (por exemplo, loops indutivos, sensores de micro-ondas, etc) por sistemas com sensores baseados em imagens, que são mais fáceis de reparar e possuem maior flexibilidade de operação. Esses sensores baseados em imagens tem de ser robustos contra situações desafiadoras que podem afetar severamente os resultados, tais como mudanças bruscas de iluminação, sombras e oclusões totais ou parciais. Neste trabalho, nossas contribuições são duas: Primeiro, propomos uma nova abordagem para detecção de sombras que combina segmentação de imagens por hipergrafos com informações de cor e textura para obtenção de resultados precisos. Segundo, um sistema de detecção de veículos que usa sensores baseados em imagens (i.e. câmeras) para detectar veículos em loops virtuais definidos pelo usuário, simulando o comportamento de um loops indutivos. Usando o método de detecção de veículos proposto, diversos loops virtuais podem ser monitorados simultaneamente por uma única câmera. Inicialmente, o fundo é modelado usando uma abordagem melhorada de modelos de misturas de Gaussianas. Então, sombras são detectadas nos loops virtuais e removidas para minimizar falsas detecção de veículos. Em seguida, veículos são detectados nos loops virtuais definidos pelo usuário usando uma combinação de bordas (obtidas por um detector eficiente) e informações de cores. O método de detecção de veículos proposto fornece um sinal de detecção que também pode ser usado para contar veículos nos loops virtuais. Os resultados experimentais mostram que nosso método proposto pode detectar e contar veículos de forma precisa (com mais de 98% de acurácia, em média), além de ser mais robusto a sombras e mudanças bruscas de iluminação que métodos comparáveis que representam o estado da arte. Além disso, experimentos realizados em datasets de detecção de sombras públicos e comuns na literatura sugerem que nosso método de detecção de sombras usando segmentação de imagens com hipergrafos é capaz de obter resultados de detecção de sombras precisos quando comparado com outros métodos recentes e relevantes.

**Palavras-chave:** Detecção de veículos. Detecção de sombras. Sistemas de monitoramento de tráfego. Loops virtuais. Sensores baseados em imagens.

## **LIST OF ABBREVIATIONS AND ACRONYMS**

GMM	Gaussian Mixture Models
RGB	Red, Green and Blue
HSV	Hue, Saturation and value
FG	Foreground
BG	Background
GW	Gabor wavelet
SGW	Simplified Gabor wavelet
NCC	Normalized Cross-Correlation

## LIST OF SYMBOLS

$\alpha$	Lower threshold of the V channel in HSV color space of color-based shadow detection
$\beta$	Upper threshold of the V channel in HSV color space of color-based shadow detection
$\tau_S$	Threshold of the S channel in HSV color space of color-based shadow detection
$\tau_H$	Threshold of the H channel in HSV color space of color-based shadow detection
$\tau_m$	Gradient magnitude threshold
$\tau_a$	Gradient direction threshold
$\tau_c$	Gradient direction correlation threshold
$K$	Number of hypergraph partitions
$\tau_\sigma$	Standard deviation threshold of the V channel in hypergraph partitions
$\tau_p$	Shadow detection sensitivity parameter
$\tau_r$	Shadow classification voting threshold
$\tau_{FG}$	Minimum number of foreground pixels to active a virtual loop
$\tau_\gamma^{\text{low}}$	Lower NCC threshold for detecting vehicles
$\tau_\gamma^{\text{high}}$	Higher NCC threshold for detecting vehicles
$\tau_l$	Minimum required number of active frames to perform histogram comparisons
$\tau_{\mathcal{H}}$	Histogram similarity threshold
$\tau_v$	Minimum required active frames to count a vehicle in a virtual loop
$\tau_z$	Binary signal similarity threshold

## LIST OF FIGURES

Figure 1.1	Genius traffic system in Thailand .....	12
Figure 1.2	Example of the virtual loops defined for three video sequences. ....	13
Figure 3.1	Summary of the proposed vehicle detection scheme.....	21
Figure 3.2	Foreground mask and background model.....	23
Figure 3.3	Shadow detection approach results.....	26
Figure 3.4	Image hypergraph partition splitting .....	28
Figure 3.5	Summary of the proposed shadow detection scheme. ....	31
Figure 3.6	The SGW kernel family.....	34
Figure 3.7	Example of edge detection using SGW kernels .....	37
Figure 3.8	Schematics of the vehicle detection process.....	42
Figure 3.9	Example of the binary signal representing passing vehicles .....	43
Figure 3.10	Median filter applied to binary signal.....	44
Figure 4.1	Shadow masks obtained by the proposed hypergraph-based shadow detection.....	47
Figure 4.2	Example of changes in camera gain .....	49
Figure A.1	Graph, hypergraph and neighborhood hypergraph example.....	67
Figure A.2	Hypergraph coarsening illustration.....	70
Figure A.3	Illustration of the 3 steps of the multilevel approach.....	71
Figure A.4	Hypergraph segmentation results.....	72



## LIST OF TABLES

Table 4.1	Information about the datasets used in the experiments.....	45
Table 4.2	Parameters used to set the hMETIS tool package for our tests. ....	48
Table 4.3	Shadow detection results for all datasets.....	53
Table 4.4	Summarizing information about the videos used in the experiments. ....	54
Table 4.5	Vehicle detection results for video sequence 1.....	55
Table 4.6	Vehicle detection results for video sequence 2.....	56
Table 4.7	Vehicle detection results for video sequence 3.....	57
Table 4.8	Vehicle detection and counting parameters used for each video sequence. ....	58
Table 4.9	Vehicle counting results for video sequence 1. ....	59
Table 4.10	Vehicle counting results for video sequence 2. ....	59
Table 4.11	Vehicle counting results for video sequence 3. ....	60

## CONTENTS

<b>1 INTRODUCTION</b> .....	<b>11</b>
<b>1.1 Thesis structure</b> .....	<b>13</b>
<b>2 RELATED WORKS</b> .....	<b>14</b>
<b>2.1 Foreground Detection</b> .....	<b>14</b>
<b>2.2 Vehicle Detection</b> .....	<b>14</b>
<b>2.3 Shadow Detection</b> .....	<b>16</b>
<b>3 PROPOSED METHOD FOR VEHICLE DETECTION IN VIRTUAL LOOPS</b> .....	<b>20</b>
<b>3.1 Foreground Detection and Background Modeling Using Gaussian Mixture Models</b>	<b>20</b>
<b>3.2 Shadow Detection and Removal</b> .....	<b>24</b>
3.2.1 Shadow Detection Using Color and Gradient Information.....	24
3.2.2 Improved Shadow Detection Using Hypergraphs.....	27
3.2.3 Shadow removal.....	32
<b>3.3 Fast Edge Detection</b> .....	<b>32</b>
<b>3.4 Vehicle Detection Using Edge and Color Information</b> .....	<b>38</b>
3.4.1 Vehicle Counting.....	42
<b>4 EXPERIMENTAL RESULTS</b> .....	<b>45</b>
<b>4.1 Hypergraph-Based Shadow Detection Results</b> .....	<b>45</b>
<b>4.2 Vehicle Detection Results</b> .....	<b>49</b>
<b>5 CONCLUSIONS AND DISCUSSION</b> .....	<b>61</b>
<b>5.1 Contributions of this Work</b> .....	<b>64</b>
5.1.1 Stochastic Shadow Detection Using a Hypergraph Partitioning Approach.....	64
5.1.2 Novel Image-based Approach for Detecting Vehicles.....	65
<b>APPENDICES</b> .....	<b>66</b>
<b>A - FUNDAMENTALS OF IMAGE SEGMENTATION USING HYPERGRAPHS</b> .....	<b>67</b>
<b>REFERENCES</b> .....	<b>73</b>

## 1 INTRODUCTION

With the development of the great urban centers, the number of vehicles is increasing every day. Therefore, traffic management systems become an important tool in the for urban development. With the recent developments in computer vision, it is now possible to use video cameras for these systems, instead of traditional sensors like ultra-sonic or microwave sensors (TIAN et al., 2011). Using traditional video cameras has several advantages over other types of sensors. With video cameras, it is possible to obtain more information about vehicles such as speed, size, trajectory and color. Also, a single video camera can analyze several traffic lanes. Another advantage is that it is possible to send the video feed itself to control stations. Finally, urban centers may have several video cameras already installed for general monitoring purposes. This allows traffic monitoring systems to use these cameras, reducing the installation costs for these systems.

The aim of traffic monitoring systems is to estimate traffic parameters, such as the number of vehicles that passed on a lane or to detect incidents (TIAN et al., 2011), such as vehicles in the opposite road direction, vehicles stopped on forbidden areas, accidents, fire/smoke, etc. Furthermore, new researches are being developed in behaviour analysis, aiming to increase drivers safety (SIVARAMAN; TRIVEDI, 2013), for example, by detecting sudden braking and reacting properly, faster than a human would react. These systems usually process videos obtained by cameras in the vehicle and also have important applications for intelligent vehicles.

The traffic parameters obtained by traffic monitoring systems have great importance in the planning and development of urban centers. They also can be used to reroute traffic and for traffic light control and optimization, improving the traffic flow in the cities. Fig. 1.1 is an example of a traffic system that indicates in real-time which streets have traffic jams (PORNANOM-CHAI; LIAMSANGUAN; VANNAKOSIT, 2008).

Traffic monitoring systems based on video cameras have advantages over other types of sensors (e.g. magnetic or microwave) (BARCELLOS et al., 2014). These camera based traffic monitoring systems can provide more information about the traffic than competing technologies (e.g. vehicles colors, size and trajectory), are easy to install and upgrade, and potentially can reduce the final monitoring system cost (BARCELLOS et al., 2014).

One of the most important step of traffic monitoring systems (and most computer vision based systems in general) is the detection of objects of interest (*i.e.* the foreground). But, several factors may negatively affect the foreground detection process, decreasing the performance of these systems. Factors like, camera noise, occlusions, cast shadows, slow or sudden changes of

Figure 1.1: Genius traffic system, in Thailand. The system indicates in real-time which streets may have traffic jams, allowing drivers to change their route and improving the traffic flow.



Source: (PORN PANOMCHAI; LIAM SANGUAN; VANNAKOSIT, 2008)

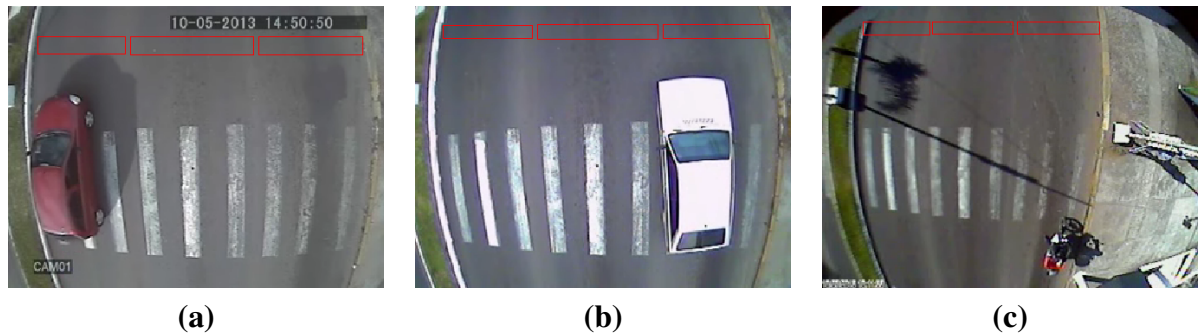
lighting in outdoors environments (*i.e.* when a cloud blocks the sun), weather conditions like rain or snow, etc. Therefore, to obtain accurate results, it is necessary to develop techniques robust against these adverse conditions.

In this work, our main contributions are: **1.** a robust scheme for vehicle detection for user defined virtual loops with high accuracy and **2.** a hypergraph-based shadow detection scheme with high shadow detection and shadow discrimination rate.

Our hypergraph-based shadow detection scheme combines color and texture information with hypergraph image segmentation to achieve accurate shadow detection results. Even though our current implementation was not suitable for real-time processing, our proposed scheme provided state-of-the-art results in common, public shadow detection datasets.

Our vehicle detection scheme does not require any previous training and it is robust against shadows, sudden and slow lighting changes. It also has low computational cost, running in real-time for our tested video sequences. Virtual loops are regions of interest defined by the user and they simulate the behaviour of traditional inductive sensors. For each lane, a virtual loop is defined and a vehicle is detected when it passes over them. Fig. 1.2 illustrates these virtual loops. Unlike many vehicle detection works that focus on counting vehicles, our proposed approach is focused on detecting vehicles in the virtual loops with the highest possible accuracy. Nonetheless, we also can use our accurate detection results to estimate the number of passing

Figure 1.2: Example of the virtual loops defined for three video sequences.



Source: Created by the author (2015).

vehicles in a lane. To the best of our knowledge, no other method that simulates an inductive loop behaviour has been proposed. This work was developed in partnership with Digicon.

## 1.1 Thesis structure

The remaining of this thesis is organized in the following way: Chapter 2 provides a brief summary of recent state-of-the-art works in the context of foreground detection, shadow detection, vehicle detection and traffic monitoring systems. Chapter 3 describes in detail our vehicle and shadow detection schemes. In Chapter 4, we describe our experiments and present our results. Finally, in Chapter 5 we present our conclusions and have a final discussion. Appendix A describes the fundamentals of image segmentation using hypergraphs, which is the basis of our proposed shadow detection scheme.

## 2 RELATED WORKS

In this chapter, we describe some recent state-of-the-art vehicle detection and counting research works. Also, since foreground and shadow detection are strongly related to our work, we describe some recent and relevant works.

### 2.1 Foreground Detection

The first step of most image-based detection schemes is the foreground detection. The method proposed by Zivkovic *et al.* (ZIVKOVIC, 2004) (ZIVKOVIC; HEIJDEN, 2006) builds upon the Gaussian Mixture Models proposed in (STAUFFER; GRIMSON, 1999) and models each pixel with a mixture of  $K$  gaussians. If a pixel does not fit into any of the gaussians of the mixture, it is considered a foreground pixel. At every step, recursive equations are used to update the mixture model (ZIVKOVIC, 2004). This method is fairly robust against noise and multi-modal backgrounds and it has a low-computational cost suitable for real-time applications.

Noh and Jeon (2013) proposed a foreground detection method that uses multiple cues: pixel texture, pixel color and region appearance. A codebook-based approach is used to cluster texture information and detect initial foreground regions. Then, an adaptive texture operator called scene adaptive local binary pattern (SALBP) is applied. The codebook scheme is also used to model color statistics. Finally, the texture and color operations are combined and the distance of the extracted blobs is matched against the edge map to obtain an accurate foreground detection. This method presents good results in challenging situations such as dynamic background and illumination changes due to the use of multiple cues.

The method proposed by St-Charles and Bilodeau (2014) builds upon the ViBe algorithm (BARNICH; DROOGENBROECK, 2011), and uses a spatio-temporal descriptor to achieve a better detection than comparable pixel-based methods. Local binary patterns occurring in multiple frames are used to improve the foreground detection. This method has good detection results, but a computational cost higher than pixel-based methods.

### 2.2 Vehicle Detection

There is a growing interest on image-based vehicle detection and some relevant recent contributions to this area are discussed next.

Sanchez *et al.* (SANCHEZ *et al.*, 2011) proposed a low cost approach for vehicle counting.

First, ‘enter’ and ‘exit’ regions are defined in the scene. Then, a background model is estimated and vehicles are detected by using background subtraction. When a vehicle is detected in an ‘enter’ region, an identification number is assigned to it and the vehicle is tracked until it passes through the ‘exit’ region. Tracking is performed by calculating the intersection between the blobs of vehicles in a frame  $T$  and frame  $T - 1$ . The speed of the vehicles is also estimated and if a vehicle becomes occluded, their position can be estimated using the vehicles’ speed until it stops being occluded or leaves the scene.

Scharcanski *et al.* (SCHARCANSKI et al., 2011) uses particle filtering to track vehicles and present an adaptive approach to handle occlusions. The method detected when a vehicle becomes occluded or leaves the occlusion, operating in two distinct modes according to the situation. In the normal mode, a Normal pdf (Probability Density Function) is used to generate a set of particles to be tracked. In the occlusion mode, a joint Normal-Rayleigh pdf is used to generate these particles. The main reason for this approach is that the Rayleigh function is oriented towards the direction with greater movement activity of a vehicle being tracked and the Normal function is oriented in the orthogonal direction, creating a ‘drift’ effect in relation to the Rayleigh function. This ‘drift’ effect moves the particles in the direction that an occlusion was detected, tracking the occluded vehicles.

Gangodkar *et al.* (GANGODKAR; KUMAR; MITTAL, 2012) proposed a vehicle segmentation method in complex conditions like adverse weather conditions (*e.g.* snow and rain), camera noise and non-adequate illumination conditions. It uses an adaptive threshold and the FSSAD (Full-search sum of absolute difference) algorithm. It also uses the notion of motion energy, which is calculated by the summation of the absolute difference between pixels in image patches. This motion energy results in good separation between dynamic backgrounds and foreground objects, allowing the detection of vehicles in the complex conditions mentioned before.

The method proposed by Yuan *et al.* (YUAN; ZHAO; WANG, 2013) performs vehicle counting in day and night environments and detect vehicles using morphological operations and color differences. A background model is estimated and updated frequently. Vehicles are detected by comparing the current frame with this background model. To decrease false detection, a secondary background model is estimated in the YCrCb color space. Using both background models, vehicles are detected when they enter or leave the region of interest. This method does not perform tracking on the vehicles.

Barcellos *et al.* (BARCELLOS et al., 2014) proposed a particle-based vehicle counting approach. Particles are grouped according to motion coherency and spatial adjacency. First,

a foreground mask is generated using Gaussian Mixture Models to determine the position in which the particles should be generated. Then, convex groups of particles are analyzed to detect vehicles. These vehicles are then tracked using color similarity in adjacent frames. The intersection of a vehicle with the virtual loops is estimated and a counter is incremented when the vehicle leaves the virtual loop region.

### 2.3 Shadow Detection

Detection of moving objects (i.e. foreground detection) is an important step in many image-based monitoring applications (CUCCHIARA et al., 2003), like traffic monitoring (SILVA; SCHARCANSKI, 2010)(SCHARCANSKI et al., 2011)(BARCELLOS et al., 2014) and people detection and/or tracking. Unfortunately, cast shadows have the same motion patterns as the objects casting them, and most foreground detection methods tend to confuse cast shadows with foreground objects (BENEZETH et al., 2010), downgrading the performance of these methods. Besides, cast shadows usually are adjacent to moving objects, and the segmentation process tends to merge foreground objects separated by cast shadows, leading to erroneous object detection and recognition (SANIN; SANDERSON; LOVELL, 2010). Therefore, the reliable detection of shadows is an important step in foreground and background detection.

Shadows occur when a light source is occluded by an object in the scene. The part of the object that is not illuminated is called self-shadow and the area projected on the scene corresponding to this occlusion is called cast shadow or moving cast shadow if the object is moving.

To evaluate the performance of shadow detection methods, (PRATI et al., 2001) proposed two metrics: the shadow detection rate and shadow discrimination rate, which are calculated as follows:

$$\eta = \frac{TP_S}{TP_S + FN_S} \text{ and } \xi = \frac{TP_F}{TP_F + FN_F}, \quad (2.1)$$

where  $TP$  and  $FN$  are the true positive and false negative pixels detected in shadow ( $S$ ) and non-shadow ( $F$ ) regions. The shadow detection rate ( $\eta$ ) increases as the number of shadow pixels detected increases. The shadow discrimination rate ( $\xi$ ) is related to a correct separation between foreground objects pixels and shadow pixels, therefore it decreases when a foreground pixel is mistakenly labeled as a shadow pixel. Usually, the average of the two measures is used as a single measure to evaluate the performance of these methods (SANIN; SANDERSON; LOVELL, 2010). Also, there is a compromise between the two measures, and by increasing the



detection rate, the discrimination rate tends to decrease and vice-versa.

A recent survey on shadow detection (SANIN et al., 2012) organizes the available methods in four categories: a) chromaticity-based methods; b) physically inspired methods; c) geometry-based methods; and d) texture-based methods.

Cucchiara *et al.* proposed a chromaticity-based method (CUCCHIARA et al., 2003) that tries to detect shadows by measuring the rate of change between the HSV components of a video frame and a background reference. This method assumes that shadow pixels in the video frame and in the background reference do not differ substantially in their hue component, and have low saturation and intensity values. However, just color information may not be able to discriminate correctly shadows and foreground objects that have darker colors similar to shadows (e.g. dark cars and their cast shadows).

Physically-based methods try to learn the appearance of shadow pixels, and may achieve higher accuracies than chromaticity-based methods (HUANG; CHEN, 2009). Unfortunately, these methods tend to fail when dealing with objects having chromaticities similar to the background (SANIN et al., 2012).

Geometry-based methods evaluate geometric features of shadows, such as the shadow orientation, size and shape (HSIEH et al., 2003). The disadvantage of these methods is that previous knowledge of the scene is required, which often is unavailable (SHABANINIA; NAGHSH-NILCHI, 2013). These methods may fail when the shadow region and the foreground object have similar orientations (SANIN et al., 2012).

Texture-based methods assume that shadows preserve most of the scene textures. For example, the method proposed in (SANIN; SANDERSON; LOVELL, 2010) uses chromaticity information to identify shadow regions, and then uses gradient information to refine the initial estimates of shadow or non-shadow pixels. This method provides good shadow detection results, and improves on the results of the chromaticity method in (CUCCHIARA et al., 2003) by combining chromaticity and gradient information. However, significant color differences between foreground objects and cast shadows are required to obtain high quality shadow detection with this method (SANIN; SANDERSON; LOVELL, 2010).

In a recent survey presented by Al-Najdawi et al. (2012), a different taxonomy was proposed to organize shadow detection methods. The shadow methods are classified based on object/environment dependency, or based on the implementation domain, which can be pixel domain or transform domain. According to (AL-NAJDAWI et al., 2012), object/environment dependent methods are designed to detect a particular type of shadow (e.g. vehicle or human cast shadows) in a specific environment (e.g. indoor or outdoor scenes). Pixel domain methods

are further divided in monochrome domain and color domain methods, and use pixel information (e.g. color) for shadow detection. Transform domain methods use different types of information for shadow detection, such as frequency domain, texture, or yet geometric information. Some transform domain methods are illustrated next.

The method proposed in Xu et al. (2005) removes insignificant cast shadows in video sequences based on edge and region information in multiple frames. A shadow is called insignificant when edges of the shadow region are not as sharp as the edges of the corresponding object. First, a mask containing moving objects and cast shadows is obtained. Then, the Canny edges detector is used to generate an edges map. The shadow regions are then removed using edge matching and region growing in multiple frames. Finally, the boundaries of the objects are improved and noise is removed by using a post-processing procedure. The motivation for this approach is that an insignificant shadow region often appears in an area where the gray levels change gradually from the background to the shadowed area. A disadvantage of this method is that if the object moves slowly, there is little change at the boundary between the object and the background, affecting negatively the results (XU et al., 2005).

Guo, Dai and Hoiem (2011) proposed a single-image shadow detection and removal method based on a paired-regions approach. First, the image is segmented using the mean-shift algorithm (COMANICIU; MEER, 2002), then a trained classifier is used to estimate the confidence that each region is a shadow region. Next, regions with the same and different illuminations are represented by a relational graph, which is partitioned using graph-cuts in shadow and non-shadow regions. Finally, the results are improved by using image matting to smooth the transitions between shadow and non-shadow regions, and shadow-free images are obtained by relighting. A disadvantage of this method is that it requires training a shadow classifier, which may involve manual labeling of shadow regions for generic scenes.

Another typical transform domain method (SHABANINIA; NAGHSH-NILCHI, 2013) handles shadow detection using the robust wavelet watershed segmentation algorithm (JUNG; SCHARCANSKI, 2005) (JUNG; SCHARCANSKI, 2004) (JUNG; SCHARCANSKI, 2002). The segmented image regions are classified as shadow and non-shadow regions using a HSV-based approach, similar to the method proposed by Cucchiara et al. (2003). A problem with this method is that using only HSV color information to classify shadow and non-shadow regions may lead to erroneous results when the foreground objects colors are similar to shadows.

The method proposed in (BULLKICH et al., 2012) uses the MTM (Matching by Tone Mapping) transformation as the distance between image patches of a video frame and a background reference. Shadowing effects are assumed to be non-linear tone mappings of the background

gray levels. Since the MTM distance is invariant to non-linear mappings between corresponding image patches in shadow and non-shadow-regions, this MTM distance results in small values for shadowed areas and in large values for foreground patches that differ from the background (BULLKICH et al., 2012). To detect shadows, the MTM transformation is applied to the spatial neighborhood of each pixel in the foreground, generating a MTM distance map. The Otsu thresholding method (OTSU, 1979) is then applied to the MTM distance map to discriminate between shadow and non-shadow regions. Unfortunately, thresholding the MTM distance map to guarantee an accurate discrimination between shadow and non-shadow regions is not trivial.

Khare, Srivastava and Khare (2014) proposed a shadow detection method based on the dual-tree complex wavelet transform to measure the difference between a video frame and the background reference in the HSV color space. The dual-tree complex wavelet transform of the difference images in the saturation and value channels is calculated. The standard deviation of the wavelet coefficients is computed, each coefficient is adaptively thresholded, and the image is reconstructed by discarding the wavelet coefficients that are associated to shadows. Finally, morphological operations are used as post-processing. This method provides interesting results, but since the wavelet coefficients are calculated based on color information only, the method performance tends to decrease if the foreground objects have chromaticities similar to shadows.

In the next chapter, we describe in detail our proposed vehicle detection and shadow detection schemes

### 3 PROPOSED METHOD FOR VEHICLE DETECTION IN VIRTUAL LOOPS

In this chapter, we describe in detail each step of our proposed vehicle detection scheme. We also describe in detail our hypergraph-based shadow detection approach. Our shadow detection approach was not suitable for real-time processing, therefore for the vehicle detection system, we decided to use a simplified approach able to perform in real-time. But, despite not being able to run in real-time, our shadow detection approach provided state-of-the-art results in public and common shadow detection datasets.

Initially, an improved Gaussian Mixture Model (ZIVKOVIC, 2004) is used to detect moving vehicles, generating a foreground mask and a background model. The foreground mask is used to detect vehicles in the virtual loop region. Shadows are detected and removed when a foreground object (vehicle) is detected in the virtual loop. After the shadows are removed, we detect the background edges using an efficient Gabor edge detector and compare the detected edges in the current frame with the background edges. If the difference is significant, a vehicle is detected in the virtual loop. Color histogram comparisons are used to improve the vehicle detection results. Fig. 3.1 shows an overview of our proposed vehicle detection scheme. Each step is detailed next.

#### 3.1 Foreground Detection and Background Modeling Using Gaussian Mixture Models

To detect the foreground objects we use the GMM approach proposed in (STAUFFER; GRIMSON, 1999) (ZIVKOVIC, 2004) (ZIVKOVIC; HEIJDEN, 2006). In this approach, each pixel is modeled by a mixture of  $K$  Gaussians, which makes this model adequate for non-static, multimodal backgrounds (BENEZETH et al., 2010).

Let  $\vec{x}^{(t)}$  denote a pixel in RGB at time  $t$ . This pixel may belong to the background (BG) or to a foreground object (FG), and it is more likely to belong to the background if:

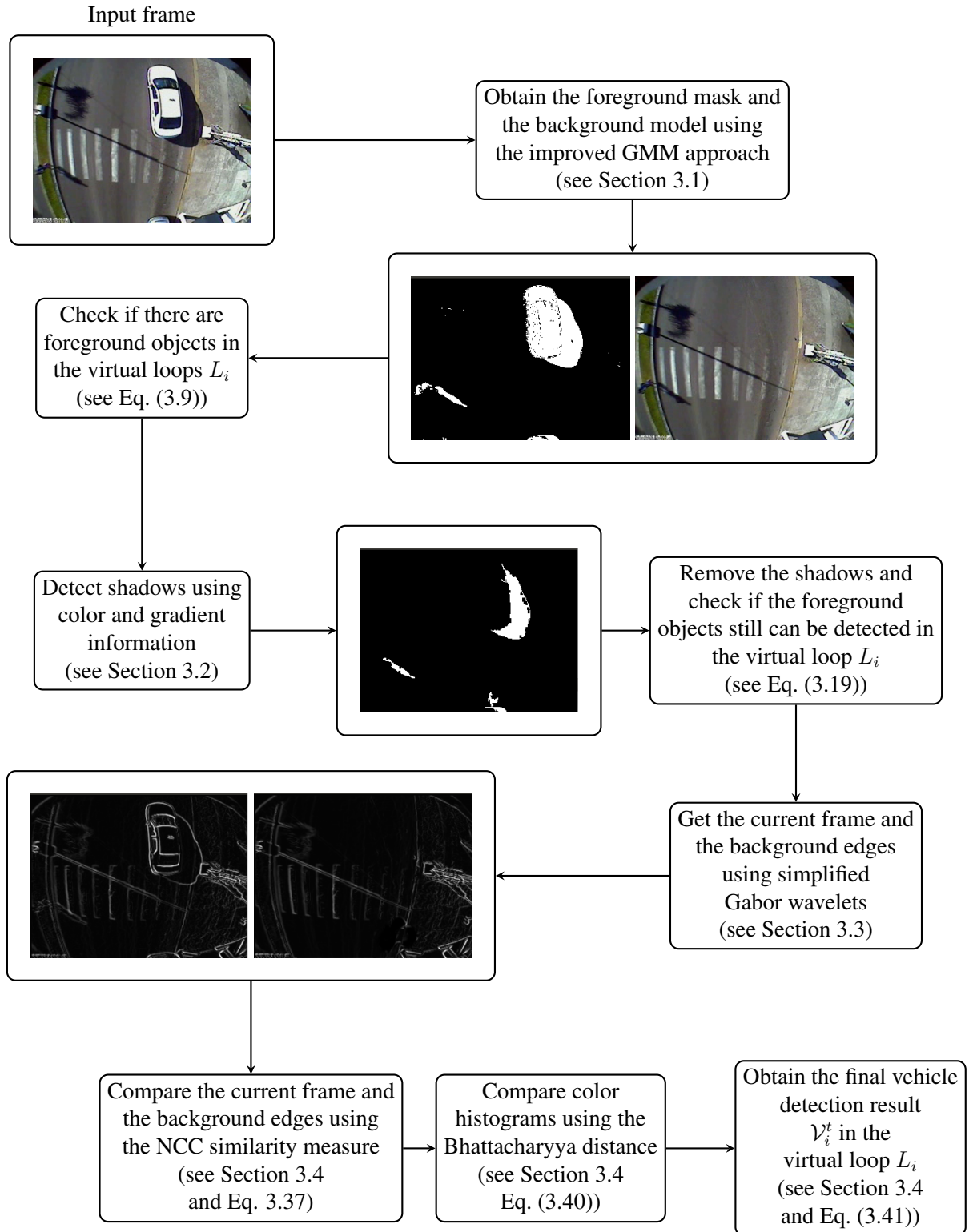
$$\frac{p(BG|\vec{x}^{(t)})}{p(FG|\vec{x}^{(t)})} = \frac{p(\vec{x}^{(t)}|BG)p(BG)}{p(\vec{x}^{(t)}|FG)p(FG)} \geq 1. \quad (3.1)$$

Since we assume no prior knowledge about the foreground objects, an uniform distribution is assumed for the appearance (likelihood) of the foreground objects  $p(\vec{x}^{(t)}|FG)$ . Therefore, a pixel belongs to the background if (ZIVKOVIC; HEIJDEN, 2006):

$$p(\vec{x}^{(t)}|BG) > c_{thr} \quad (3.2)$$

where  $c_{thr} = p(\vec{x}^{(t)}|FG)p(FG)/p(BG)$  is adopted as a threshold value and  $p(\vec{x}^{(t)}|BG)$  is the

Figure 3.1: Summary of the proposed vehicle detection scheme.



Source: Created by the author (2015).

estimated background model. A training set  $\chi$  is used to estimate the background model, which is denoted by  $\hat{p}(\vec{x}|\chi, BG)$ . In order to adapt to changes, like the illumination gradually changing in an outdoor scene or new objects coming into the scene, the training set is updated by adding new samples and discarding old ones (ZIVKOVIC; HEIJDEN, 2006). Therefore, a reasonable adaptation period  $T$  is chosen, and at time  $t$  we have the training set  $\chi_T = \{x^{(t)}, \dots, x^{(t-T)}\}$ , and  $\chi_T$  is updated for each new sample  $x^{(t)}$ . The estimated training set density  $\hat{p}(\vec{x}|\chi_T, BG + FG)$  with  $M$  components is given by:

$$\hat{p}(\vec{x}|\chi_T, BG + FG) = \sum_{m=1}^M \hat{\pi}_m \mathcal{N}(\vec{x}; \hat{\mu}_m, \hat{\sigma}_m^2 I), \quad (3.3)$$

where  $\hat{\mu}_1, \hat{\mu}_2, \dots, \hat{\mu}_M$  are the estimates of the means and  $\hat{\sigma}_1^2, \hat{\sigma}_2^2, \dots, \hat{\sigma}_M^2$  are the estimates of the variances of the  $M$  Gaussian components and  $\mathcal{N}$  is a Gaussian PDF. The mixing weights denoted by  $\hat{\pi}_m$  are non-negative and add up to 1 (ZIVKOVIC; HEIJDEN, 2006). Given a new data sample  $\vec{x}^{(t)}$  at time  $t$ , the model is updated according to the following recursive equations (ZIVKOVIC; HEIJDEN, 2006):

$$\hat{\pi}_m^{(t)} \leftarrow \hat{\pi}_m^{(t-1)} + \alpha(o_m^{(t)} - \hat{\pi}_m^{(t-1)}), \quad (3.4)$$

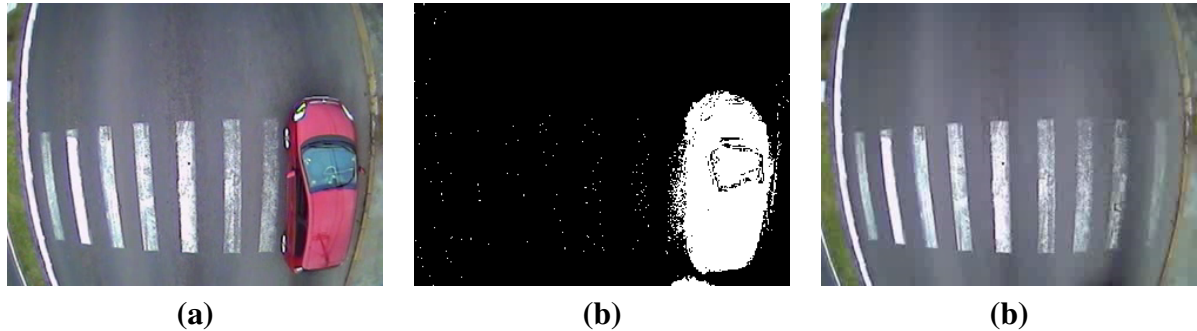
$$\hat{\mu}_m^{(t)} \leftarrow \hat{\mu}_m^{(t-1)} + o_m^{(t)}(\alpha/\hat{\pi}_m^{(t)})\vec{\delta}_m, \quad (3.5)$$

$$\hat{\sigma}_m^{2(t)} \leftarrow \hat{\sigma}_m^{2(t-1)} + o_m^{(t)}(\alpha/\hat{\pi}_m^{(t)})(\vec{\delta}_m^T \vec{\delta}_m - \hat{\sigma}_m^{2(t-1)}), \quad (3.6)$$

where  $\vec{\delta}_m = \vec{x}^{(t)} - \hat{\mu}_m^{(t)}$ ,  $o_m^{(t)}$  is the ownership of the new sample and  $\alpha$  is a constant used to limit the influence of old data ( $\alpha = 1/T$  in our experiments). The ownership  $o_m^{(t)}$  of a new sample is set to ‘1’ for the ‘closest’ mixture component with the largest weight  $\hat{\pi}_m$  and it is set to ‘0’ for the other components. A sample is ‘close’ to a mixture component if its squared distance to the component mean is less than  $k$  (e.g.  $k = 3$ ) standard deviations ( $\sigma_m$ ) (ZIVKOVIC, 2004). The squared distance to the  $m$ -th mixture component is calculated as  $D_m^2(\vec{x}^{(t)}) = \vec{\delta}_m^T \vec{\delta}_m / \hat{\sigma}_m^2$ . If no ‘close’ component is found, a new mixture component is generated with  $\hat{\pi}_{M+1}^{(t)} = \alpha$ ,  $\hat{\mu}_{M+1}^{(t)} = \vec{x}^{(t)}$  and  $\hat{\sigma}_{M+1}^{(t)} = \sigma_0$ , where  $\sigma_0$  is some appropriate initial standard deviation. If the maximum number of components  $M$  is reached, the one with the smallest  $\hat{\pi}_m$  is discarded to accommodate the new component.

Foreground objects usually are represented by mixture components with smaller weights  $\hat{\pi}_m$ , then the background model can be approximated by selecting the first  $b$  mixture components

Figure 3.2: Illustration of the resulting foreground mask and background model: (a) original frame; (b) foreground mask  $F_G$ ; and (c) background model  $B$ .



Source: Created by the author (2015).

with the largest weights (ZIVKOVIC, 2004):

$$p(\vec{x}|\mathcal{X}_T, BG) \sim \sum_{m=1}^b \hat{\pi}_m \mathcal{N}(\vec{x}; \hat{\mu}_m, \sigma_m^2 I). \quad (3.7)$$

If the components are sorted to have decreasing weights  $\hat{\pi}_m$ , then we have:

$$BG = \arg \min_b \left( \sum_{m=1}^b \hat{\pi}_m > (1 - c_f) \right), \quad (3.8)$$

where  $c_f$  is a measure of the maximum portion of the scene data that should be assigned to foreground objects. At the end of this process we obtain a background model  $BG$ , and a binary foreground mask  $F_G$  in which pixels that belong to foreground objects have the value ‘1’, or ‘0’ otherwise. Fig. 3.2 illustrates the resulting foreground mask and background model. If the number of foreground pixels in a user-defined virtual loop is higher than a given threshold  $\tau_{FG}$ , the vehicle detection process is started. More formally, let  $L_i$  be the  $i$ -th user-defined virtual loop,  $F_G(x, y)$  the foreground mask value of a pixel in position  $(x, y)$  and  $\mathcal{V}_i^t$  the result of the vehicle detection in the  $i$ -th virtual loop at a time  $t$ , such that  $\mathcal{V}_i^t = 1$  if a vehicle is detected in the virtual loop  $L_i$ , and  $\mathcal{V}_i^t = 0$  otherwise. Additionally, to guarantee that at least  $\tau_{FG}$  foreground pixels have been detected in the virtual loop  $L_i$ , the following check is performed:

$$\mathcal{V}_i^t = 0 \text{ if } \sum_{(x,y) \in L_i} F_G(x, y) < \tau_{FG}. \quad (3.9)$$

The foreground mask  $F_G$  may contain vehicles cast shadows, which are artifacts that could incorrectly activate a virtual loop. Therefore, if Eq. (3.9) is not satisfied, the next step of our vehicle detection scheme is shadow detection.

### 3.2 Shadow Detection and Removal

The foreground mask  $F_G$  (Section 3.1) may contain cast shadows pixels. Therefore, to obtain accurate vehicle detection results we detect and remove the shadows.

To perform shadow detection we use color and gradient information (SANIN; SANDERSON; LOVELL, 2010). Cucchiara et al. (2003) proposed using color information to detect shadows, since shadow regions should maintain most of the original color, but tend to have lower luminance values. Therefore, the rate of change between the color components of the current frame  $F$  and the background model  $B$  is measured in the HSV color space. However, only color information may be insufficient to detect shadows when the foreground objects and the shadows have similar color. Sanin, Sanderson and Lovell (2010) improved the method proposed in (CUCCHIARA et al., 2003) by combining color information and texture gradient information, since shadows tend to preserve most of the background textures. Consequently, by combining both color and gradient information, it is possible to obtain higher discrimination rates between shadows and their respective foreground objects.

Also, we improved on the method proposed by Sanin, Sanderson and Lovell (2010) by combining color and texture information with hypergraph image segmentation. This shadow detection scheme provided state-of-the-art results on our experiments in shadow detection datasets, but was not suitable for real-time processing. Therefore, we decided to use the color and gradient-based shadow detection for the vehicle detection system.

In this section, we describe both the color and gradient shadow detection and our improved hypergraph image segmentation scheme.

#### 3.2.1 Shadow Detection Using Color and Gradient Information

The first step of the shadow detection stage is to use color information to create a rough estimate of the shadow regions. A pixel  $p$  is in this rough shadow estimate if (SANIN; SANDERSON; LOVELL, 2010):

$$\begin{cases} \alpha \leq (F_p^V / B_p^V) \leq \beta, \text{ and} \\ (F_p^S - B_p^S) \leq \tau_S, \text{ and} \\ \|F_p^H - B_p^H\| \leq \tau_H, \end{cases} \quad (3.10)$$

where  $F = \{F^H, F^S, F^V\}$  and  $B = \{B^H, B^S, B^V\}$  are the color components in HSV color space, and  $\alpha \in [0, 1], \beta \in [0, 1], \tau_S \in [0, 255]$  and  $\tau_H \in [0, 255]$  are thresholds defined experi-



mentally. In this step, the thresholds are set to values so that a large number of pixels is included in this initial shadow regions estimate, which may later contribute to increase the shadow detection rate. In our experiments, we used the following values:  $\alpha = 0.21$ ,  $\beta = 0.99$ ,  $\tau_S = 76$  and  $\tau_H = 93$ .

Next, the connected shadow components (shadow pixels blobs) are extracted from the shadow mask using Eq. (3.10). Each connected component  $R_i$  is a candidate shadow region, and the gradient magnitude  $|\nabla_p|$  and gradient direction  $\theta_p$  are calculated at each pixel  $p(x, y) \in R_i$  as follows:

$$|\nabla_p| = \sqrt{\nabla_x^2 + \nabla_y^2}, \quad (3.11)$$

$$\theta_p = \text{atan2}(\nabla_y/\nabla_x). \quad (3.12)$$

where  $\nabla_x$  and  $\nabla_y$  are the gradient magnitudes in the  $x$  and  $y$  directions, respectively.

The next step is to calculate the difference between the gradient directions in the frame  $F$  and background model  $B$ . To reduce the effect of noise, only the pixels with sufficient gradient magnitude (*i.e.*  $|\nabla_p| \geq \tau_m \in [0, 10]$ ) are considered (SANIN; SANDERSON; LOVELL, 2010). In our experiments, a value of  $\tau_m = 6$  was adequate. This gradient direction difference is calculated as follows:

$$\Delta\theta_p = \arccos \left[ \frac{\nabla_x^F \nabla_x^B + \nabla_y^F \nabla_y^B}{\{(\nabla_x^{F^2} + \nabla_y^{F^2})(\nabla_x^{B^2} + \nabla_y^{B^2})\}^{\frac{1}{2}}} \right]. \quad (3.13)$$

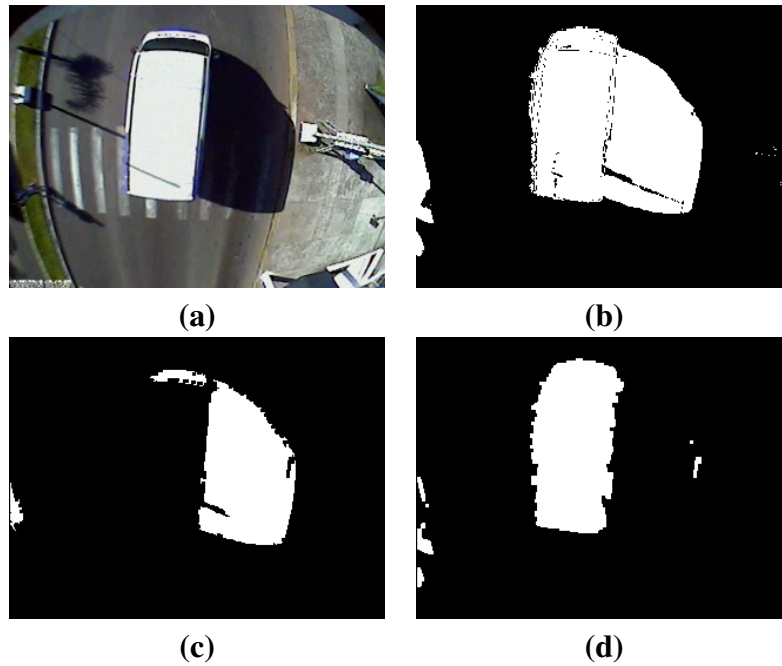
Finally, the correlation between the gradient directions in the frame  $F$  and background model  $B$  is estimated for each region  $R_i$  by:

$$c^{R_i} = \left\{ \sum_{p=1}^n H(\tau_a - \Delta\theta_p) \right\} / n, \quad (3.14)$$

where  $n$  is the number of pixels in  $R_i$ , and  $H(\cdot)$  is the unit step function which evaluates to ‘1’ if  $\Delta\theta_p \leq \tau_a \in [0, \pi]$ , and to ‘0’ otherwise,  $c^{R_i}$  is the fraction of pixels in  $R_i$  with gradient directions similar in the frame  $F$  and in the background model  $B$ . If  $c^{R_i} \geq \tau_c \in [0, 1]$ , then the region  $R_i$  is considered a shadow region. In our experiments, we used  $\tau_a = \pi/10$  and  $\tau_c = 0.2$ . At the end of this process, we obtain a shadow mask  $\mathcal{S}$ . Fig. 3.3 shows the results of the shadow detection scheme for a given video frame.

At the end of this process we obtain a shadow mask  $\mathcal{S}$ , which is used to remove the vehicle’s cast shadows.

Figure 3.3: Illustration of the results obtained with the proposed shadow detection approach: **(a)** original frame with cast shadows; **(b)** foreground model obtained with the GMM background modeling approach; **(c)** shadow mask obtained using color and gradient information (SANIN; SANDERSON; LOVELL, 2010), and **(d)** foreground mask obtained after shadow removal.



Source: Created by the author (2015).

### 3.2.2 Improved Shadow Detection Using Hypergraphs

We build on the shadow detection approach described in Section 3.2.1 by performing hypergraph image segmentation and initially performing a rough classification in the resulting image partitions as shadows or non-shadows partitions. Then, we combine this initial classification with the color and gradient shadow detection to validate (or correct) our initial classification label. This is performed as follows:

Initially, we pre-process an input frame  $F$  to improve the contrast between shadow and non-shadow regions by normalizing its  $V$  component in the HSV color space. More formally, let  $F = \{H^F, S^F, V^F\}$  be the HSV components of the image  $F$ , the normalized  $V$  component can be obtained by:

$$V_m^F = \frac{V^F - \min(V^F)}{\max(V^F) - \min(V^F)}, \quad (3.15)$$

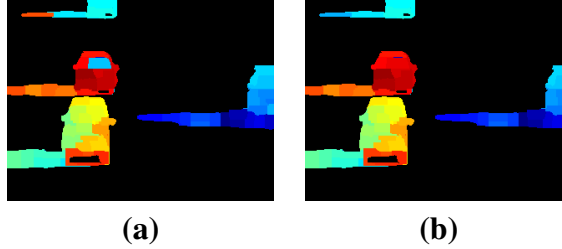
where  $V_m^F$  is the normalized  $V$  component and  $V^F$  is the original value of the  $V$  of a frame  $F$ . Therefore, the pre-processed image  $F_m$  is obtained by substituting the original  $V$  component by its normalized version (*i.e.*  $F_m = \{H^F, S^F, V_m^F\}$ ).

Then, we obtain initial shadow estimates using color and gradient information as described in Section 3.2.1. Let  $x_{p(i,j)}$  be the label of a pixel obtained with color and gradient information as described in Section 3.2.1, such that:  $x_{p(i,j)} = s$  if  $p(i,j)$  is a shadow pixel, or  $x_{p(i,j)} = \bar{s}$  otherwise. These labels are going to be validated (or changed) later to obtain the final shadow mask.

We perform a hypergraph image segmentation on the detected foreground objects and obtain  $K$  partitions (for details about hypergraph image segmentation, please refer to Appendix A). The number  $K$  of partitions is user-defined and each partition may contain more than one group of foreground pixels (blob). Each partition is checked for this situation and partitions that contain more than one blob are split until each partition contains only one blob. This splitting process may result in  $K + L$  partitions, where  $L$  is the number of additional image hypergraph partitions created by the splitting process. An illustration of this splitting process can be seen in Fig. 3.4.

Then, we classify all pixels in each image hypergraph partition  $\mathcal{P}_n$  as a shadow (or non-

Figure 3.4: Example of the image hypergraph partition splitting process: **(a)** partitions consisting of more than one blob (orange and cyan partitions); **(b)** after the splitting process, all partitions contain only one blob of foreground pixels.



Source: Created by the author (2015).

shadow) pixels according to the following criterion:

$$x_{\mathcal{P}_n} = \begin{cases} s & \text{if } \mu_{\mathcal{P}_n}^F \leq \mu_{\mathcal{P}_n}^B \text{ and } \sigma_{\mathcal{P}_n} \leq \tau_\sigma, \\ \bar{s}, & \text{otherwise;} \end{cases} \quad (3.16)$$

where  $x_{\mathcal{P}_n}$  is the label of the pixels in the hypergraph partition  $\mathcal{P}_n$ ,  $\mu_{\mathcal{P}_n}^F$  and  $\mu_{\mathcal{P}_n}^B$  are the  $V$  channel average of the HSV color pixels  $p \in \mathcal{P}_n$  in the frame  $F$  and in the background reference  $B$ , respectively, and  $\sigma_{\mathcal{P}_n}$  is the standard deviation of the  $V$  channel of the HSV color pixels  $p \in \mathcal{P}_n$ . The reasoning is that pixels belonging to shadow regions tend to be darker in the current frame than in the background reference (which contains no shadows), therefore if the foreground pixels are brighter in average (i.e.  $\mu_{\mathcal{P}_n}^F > \mu_{\mathcal{P}_n}^B$ ), no further validation is required and all pixels in  $\mathcal{P}_n$  are classified as non-shadow pixels. Also, image shadow regions tend to be more uniform in the  $V$  channel than foreground regions, so the pixels in an image hypergraph partition  $\mathcal{P}_n$  that have large intensity variability in average (i.e.  $\sigma_{\mathcal{P}_n} > \tau_\sigma$ ) are classified as non-shadow pixels. In our experiments (see Section 4.1), we used  $\tau_\sigma = 0.15$ .

Let  $\mathcal{X}_{p(i,j)}$  be the final label for a pixel  $p(i,j)$ . If the set of pixels in an image hypergraph partition  $\mathcal{P}_n$  is classified as a non-shadow (i.e.  $x_{\mathcal{P}_n} = \bar{s}$ ) then  $\forall p(i,j) \in \mathcal{P}_n$  the pixels labels in  $\mathcal{P}_n$  are set to  $\mathcal{X}_{p(i,j)} = \bar{s}$  and the pixels in  $\mathcal{P}_n$  are non-shadow pixels and do not need further validation. On the other hand, an image hypergraph partition  $\mathcal{P}_n$  classified as a shadow region (i.e.  $x_{\mathcal{P}_n} = s$ ) may contain some pixels labeled as non-shadow pixels (i.e.  $x_{p(i,j)} = \bar{s}$ ), so the next step in the proposed shadow detection method is to validate the individual shadow or non-shadow pixels labels using the image hypergraph partition labels ( $x_{p(i,j)}$  and  $x_{\mathcal{P}_n}$ ), obtaining the final shadow or non-shadow label  $\mathcal{X}_{p(i,j)}$  for the pixels  $p(i,j) \in \mathcal{P}_n$  forming a shadow mask for the image.

The shadow pixels classification based on color and gradient information (see Section 3.2.1) are validated by the hypergraph partitions detected as shadow regions using a stochastic majority rule, and this process generates the final shadow mask, as described next. This is motivated by the fact that at this stage some of the shadow pixels obtained with Eqs. (3.10), (3.13) and (3.14) may be outliers or incorrectly labeled.

The image hypergraph segmentation approach generates image segments (hypergraph partitions) consisting of pixels that are visually similar and spatially close (see Appendix A), so these image segments are color homogeneous. If most pixels in an image region  $\mathcal{P}_n$  are shadow pixels (or non-shadow pixels), then all pixels in the region  $\mathcal{P}_n$  are classified as shadow pixels (or non-shadow pixels). Therefore, pixels that have been estimated previously as shadow pixels are validated (or relabeled as non-shadow pixels) by considering the label of similar pixels in their neighborhoods, resulting in an improved shadow/non-shadow discrimination. This refinement of the shadow detection process is described in more detail below.

The proportions of shadow/non-shadow pixels are estimated in each image hypergraph partition  $\mathcal{P}_n$  labeled as a shadow region (i.e.  $x_{\mathcal{P}_n} = s$ ) (see Eq. (3.16)). If  $\mathcal{P}_n$  has more shadow pixels than non-shadow pixels, then  $\mathcal{P}_n$  is validated as a shadow region, and the final shadow mask labels  $\mathcal{X}_{p(i,j)}$  of the pixels  $p(i,j) \in \mathcal{P}_n$  are obtained as follows:

$$\forall p(i,j) \in \mathcal{P}_n, \mathcal{X}_{p(i,j)} = \begin{cases} s, & \text{if } \frac{\sum_{p(i,j) \in \mathcal{P}_n} \delta(x_{p(i,j)}, s)}{\mathcal{N}} \\ & \geq \frac{\sum_{p(i,j) \in \mathcal{P}_n} \delta(x_{p(i,j)}, \bar{s})}{\mathcal{N}} - \tau_p \\ & \text{and } x_{\mathcal{P}_n} = s, \\ \bar{s}, & \text{otherwise,} \end{cases} \quad (3.17)$$

where  $p(i,j)$  is a pixel belonging to a partition  $\mathcal{P}_n$ ,  $\mathcal{X}_{p(i,j)}$  is the final shadow mask label of  $p(i,j) \in \mathcal{P}_n$ ,  $\mathcal{N}$  is the total number of pixels in  $\mathcal{P}_n$ ;  $x_{p(i,j)} = s$  if the pixel has been classified as a shadow pixel using Eq. (3.10) and  $c^{\mathcal{R}_m} \geq \tau_c$  (see Eq. (3.14)) and  $x_{p(i,j)} = \bar{s}$  otherwise (see Section 3.2.1);  $\delta(a,b)$  is the Kronecker delta such that  $\delta(a,b)$  evaluates to 1 if  $a = b$  or to 0 otherwise; and  $\tau_p$  is a threshold. In our experiments  $\tau_p = 0.24$  (see Section 4.1).

The multilevel hypergraph partitioning algorithm is non-deterministic due to the random bisection used in the initial partitioning phase (KARYPIS et al., 1999) (see Appendix A and Fig. A.3), therefore the multiple partitionments obtained for the same hypergraph may differ, leading to different shadow mask labels  $\mathcal{X}_{p(i,j)}$  for the same pixels. Therefore, a stochastic majority voting scheme is used as a final step to obtain the shadow mask pixels labels  $\hat{\mathcal{X}}_{p(i,j)}$ .

In this stochastic voting scheme,  $N$  hypergraph partitionments are obtained stochastically (see Appendix A), and  $N$  classification labels  $\mathcal{X}_{p(i,j)}$  are generated for each pixel  $p(i,j)$ . Let  $\mathcal{X}_{p(i,j)}^l$  be the shadow mask label of a pixel  $p(i,j)$  (see Eq. (3.17)) obtained in the  $l$ -th hypergraph partitionment. The final shadow mask  $\hat{\mathcal{X}}_{p(i,j)}$  contains the pixels  $p(i,j)$  that were classified as shadows in at least  $\tau_r \in [0, N]$  hypergraph partitionments, as indicated below:

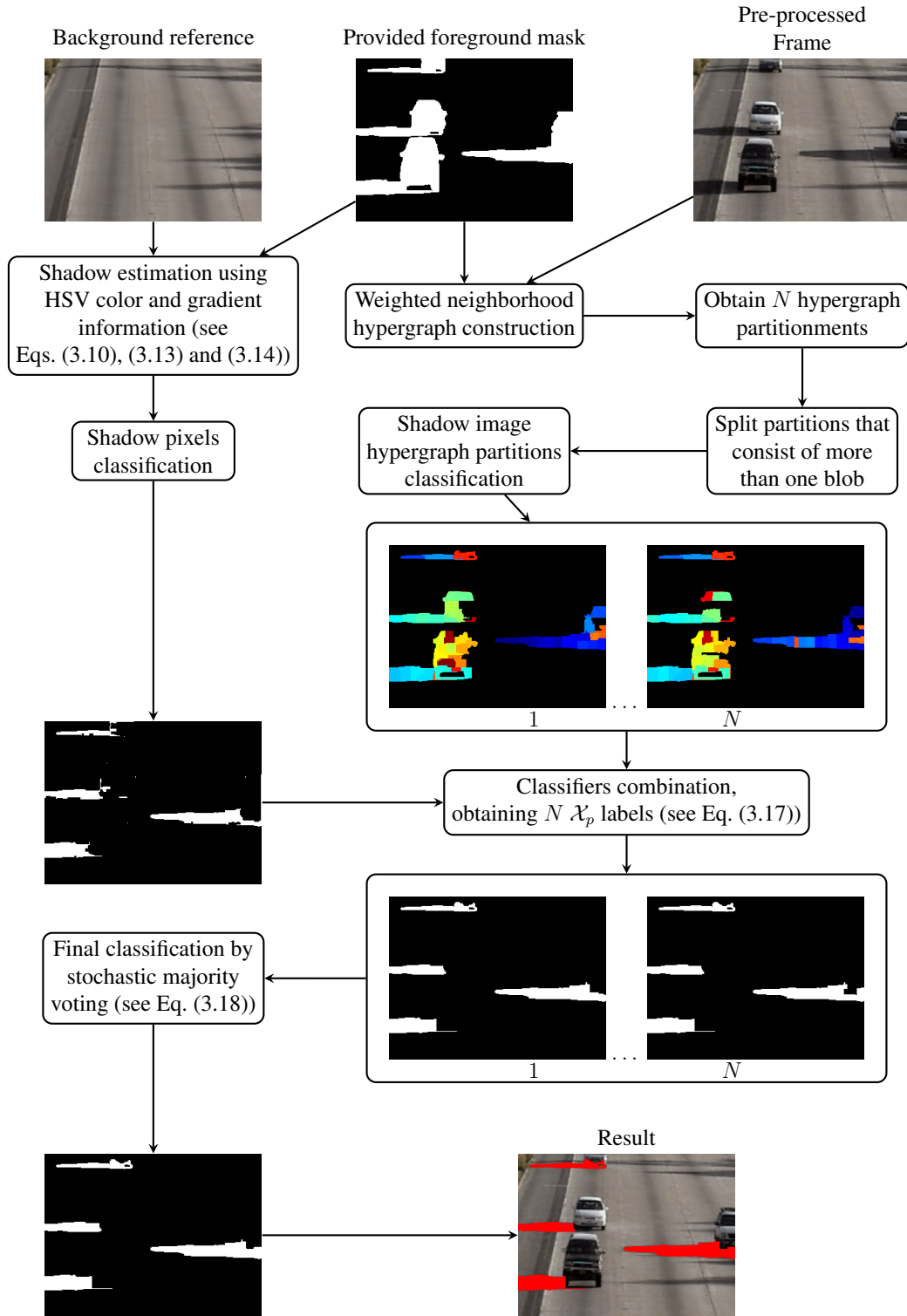
$$\begin{cases} \mathcal{X}_{p(i,j)} = 1 \text{ if } \left\{ \sum_{l=1}^N \delta(\mathcal{X}_{p(i,j)}^l, s) \geq \tau_r \right\}, \\ \mathcal{X}_{p(i,j)} = 0 \text{ otherwise.} \end{cases}, \quad (3.18)$$

where  $\delta(a, b)$  is the Kronecker delta. In our experiments, we used  $\tau_r = 3$  and  $N = 10$ .

A summary of the proposed hypergraph-based shadow detection scheme is shown in Fig. 3.5. Also, our hypergraph-based shadow detection scheme is summarized below, step by step:

1. Build the weighted hypergraph of the video frame in the LAB color space, using the DeltaE color distance (see Eq. (A.6)) and the provided foreground mask;
2. Obtain  $K$  image hypergraph partitions  $N$  times, and for each different partitionment do:
  - (a) Split the partitions  $\mathcal{P}_n$  that contain more than one blob of pixels;
  - (b) Classify pixels  $p(i,j)$  by using the color (see Eq. (3.10)) and gradient correlation information (i.e.  $c^{\mathcal{R}^m} \geq \tau_c$ , where  $c^{\mathcal{R}^m}$  is given by Eq. (3.14)) as shadow (or non-shadow) pixels (i.e.  $x_{p(i,j)} = s$  or  $x_{p(i,j)} = \bar{s}$ );
  - (c) Classify each partition  $\mathcal{P}_n$  by using its mean ( $\mu_{\mathcal{P}_n}^F, \mu_{\mathcal{P}_n}^B$ ) and its standard deviation ( $\sigma_{\mathcal{P}_n}$ ) of the  $V$  channel of the HSV color representation as a shadow (or a non-shadow) partition (i.e.  $x_{\mathcal{P}_n} = s$  or  $x_{\mathcal{P}_n} = \bar{s}$ ), according to Eq. (3.16);
  - (d) Combine the classification labels obtained in (b) and (c) by calculating the proportion of shadow pixels in each shadow hypergraph partition  $\mathcal{P}_n$ . If a partition  $\mathcal{P}_n$  contains more shadow/non-shadow pixels, classify all pixels  $p(i,j) \in \mathcal{P}_n$  as shadow/non-shadow pixels, obtaining the  $l$ -th pixel label  $\mathcal{X}_{p(i,j)}^l$  according to Eq. (3.17);
3. Obtain the final shadow mask  $\hat{\mathcal{X}}_{p(i,j)}$  containing all pixels  $p(i,j)$  that were classified as shadow pixels (i.e.  $\mathcal{X}_{p(i,j)}^l = s$ ) in at least  $\tau_r$  of the  $N$  partitionments, according to Eq. (3.18).

Figure 3.5: Summary of the proposed shadow detection scheme.



Source: Created by the author (2015).

### 3.2.3 Shadow removal

After obtaining the shadow mask  $\mathcal{S}$  (see Section 3.2.1), another check on the virtual loops  $L_i$  is performed to verify if the pixels previously detected as foreground in  $F_G$  and also are in  $\mathcal{S}$  (see Eq. (3.9)) actually are shadow pixels. More formally, let  $F'_G = F_G - \mathcal{S}$  be the estimated foreground mask, then a virtual loop  $L_i$  is not activated if it does not contain vehicles and contain only shadows, which is verified by:

$$\mathcal{V}_i^t = 0, \text{ if } \sum_{(x,y) \in L_i} F'_G(x,y) < \tau_{F_G}, \quad (3.19)$$

where  $\mathcal{V}_i^t$  is the vehicle detection result for the virtual loop  $L_i$  in the time instant  $t$ ,  $F'_G(x,y)$  denotes a pixel belonging to the estimated foreground mask  $F'_G$  without shadows, and  $\tau_{F_G}$  is the same threshold used in Eq. (3.9). If equation Eq. (3.19) is satisfied, it means that the foreground pixels detected on the virtual loop  $L_i$  are shadow pixels and no further processing is performed for this virtual loop. Otherwise, we proceed to the next step in our proposed vehicle detection scheme (see Fig. 3.1).

### 3.3 Fast Edge Detection

The next step of our proposed vehicle detection method consists of detecting the edges in the frame  $F$  and background model  $B$ . These edges will be compared to check if the foreground object in the virtual loop  $L_i$  in fact is a vehicle. To perform the edge detection, we use simplified Gabor wavelets (JIANG; LAM; SHEN, 2009), since this approach can generate accurate results efficiently and is also highly parallelizable, improving the efficiency of the method.

The human visual system can be viewed as being composed of a filter bank and these filters responses can be modeled using Gabor functions of different frequencies and orientations (JIANG; LAM; SHEN, 2009). In the spatial domain, a 2-D Gabor filter is a Gaussian kernel function modulated by a sinusoid (JIANG; LAM; SHEN, 2009):

$$G(x,y) = \exp \left[ -\frac{x^2 + y^2}{2\sigma^2} \right] \exp [j\omega(x\cos\theta + y\sin\theta)], \quad (3.20)$$

where  $\sigma$  is the standard deviation of the Gaussian function along the  $x$  and  $y$  directions, and  $\omega$  denotes the spatial frequency. The Gabor wavelets respond strongly to an edge if the edge direction is perpendicular to the wave vector  $(\omega\cos\theta, \omega\sin\theta)$ , and by measuring the response magnitude, local properties of the images can be effectively estimated (JIANG; LAM; SHEN,



2009).

Pellegrino, Vanzella and Torre (2004) proposed to use the imaginary part of a Gabor filter as an efficient and robust edge detector. The imaginary part of a Gabor wavelet is defined as follows (JIANG; LAM; SHEN, 2009):

$$S(x, y) = \exp \left[ -\frac{x^2 + y^2}{2\sigma^2} \right] \sin [\omega(x\cos\theta + y\sin\theta)]. \quad (3.21)$$

To detect multiple types of edges in the image,  $\theta$  is set to 8 different orientations (*i.e.*  $\theta = k\pi/8, k = 0, \dots, 7$ ), and  $\omega \cdot \sigma \leq 1$  (JIANG; LAM; SHEN, 2009). Since the extraction of these Gabor features is computationally expensive, a simplified version of these Gabor wavelets (GWs) is used instead (JIANG; LAM; SHEN, 2009). Next, this simplified approach is described in detail.

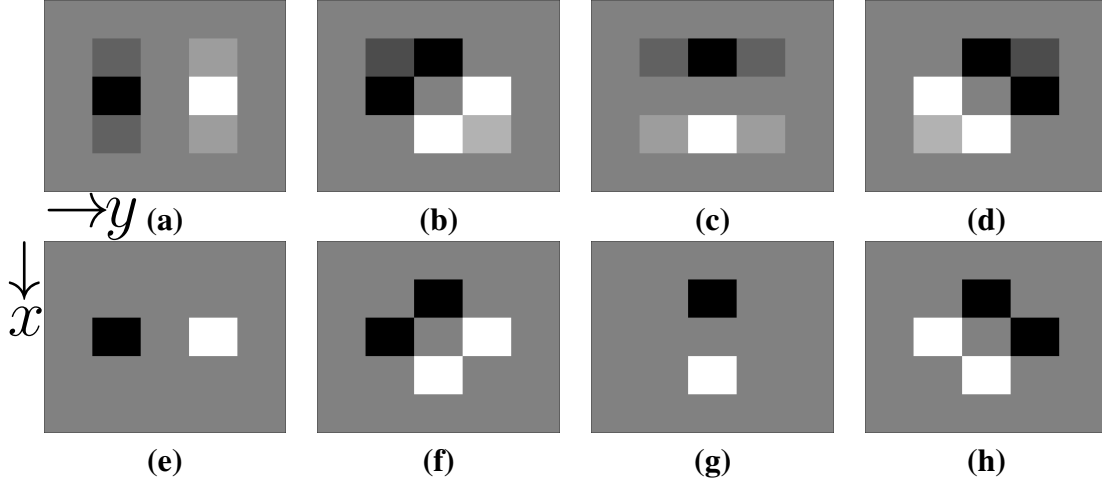
First, the Gabor wavelet is quantized into a number of gray levels. The number of gray levels of the simplified Gabor wavelet (SGW) depends on the number of quantization levels. If more levels are used, the SGW will be more similar to the original GW, but the computational cost is increased, therefore there is a trade-off between computational effort and approximation accuracy. To determine the number of quantization levels the approach proposed in (CHOI et al., 2008) is used. One of the quantization levels in the SGW is set to zero. The SGW is antisymmetric, therefore the number of quantization levels of the positive and negative values are equal and denoted as  $n_l$  (JIANG; LAM; SHEN, 2009). Let  $A$  be the largest magnitude of a GW, then the corresponding quantization levels for positive levels  $q_+(k)$  and negative levels  $q_-(k)$  are defined as:

$$q_+(k) = \frac{A}{2n_l+1} \cdot 2k \quad , \quad q_-(k) = -\frac{A}{2n_l+1} \cdot 2k, \quad (3.22)$$

where  $k = 1, \dots, n_l$ . By selecting different center frequencies and orientations, we can generate a family of GW kernels using Eq. (3.21), and then quantize them according to Eq. (3.22), obtaining a family of SGWs. Fig. 3.6 illustrates the results for a family of SGWs.

By convolving an image  $I$  with each of the GWs kernels generated with equation Eq. (3.21) we can extract the edge features. This convolution can be efficiently calculated by using the FFT (Fast Fourier Transform) algorithm, and finally the inverse FFT (IFFT) algorithm (JIANG; LAM; SHEN, 2009). Concatenating the outputs of each of the SGWs kernels, we obtain a

Figure 3.6: The SGW kernel family generated with **(a)**  $\omega = 0.3\pi$  and  $\theta = 0$ ; **(b)**  $\omega = 0.3\pi$  and  $\theta = \pi/4$ ; **(c)**  $\omega = 0.3\pi$  and  $\theta = \pi/2$ ; **(d)**  $\omega = 0.3\pi$  and  $\theta = 3\pi/4$ ; **(e)**  $\omega = 0.5\pi$  and  $\theta = 0$ ; **(f)**  $\omega = 0.5\pi$  and  $\theta = \pi/4$ ; **(g)**  $\omega = 0.5\pi$  and  $\theta = \pi/2$ ; **(h)**  $\omega = 0.5\pi$  and  $\theta = 3\pi/4$ ;



Source: Created by the author (2015).

feature vector  $\phi_{\omega,\theta}$ , of size  $N_W \cdot N_H$ :

$$\phi_{\omega,\theta} = [\phi_{\omega,\theta}(0, 0), \phi_{\omega,\theta}(0, 1), \dots, \phi_{\omega,\theta}(0, N_H - 1), \phi_{\omega,\theta}(1, 0), \dots, \phi_{\omega,\theta}(N_W - 1, N_H - 1)]^T, \quad (3.23)$$

where  $T$  denotes the transpose, and  $N_W$  and  $N_H$  are the width and height of an image  $I$ . Even if the FFT algorithm is used to reduce the computational cost, this operation still is computationally expensive. For  $n_\omega$  different scales and  $n_\theta$  different orientations, the number of SGWs involved is  $n_\omega \cdot n_\theta$  (JIANG; LAM; SHEN, 2009), therefore we use the simplified Gabor wavelets family in a more efficient manner, as described next.

Let  $\phi'_{\omega,\theta}(x, y)$  be an SGW local edge feature of an image  $I$  extracted by the SGW with scale  $\omega$  and orientation  $\theta$ . The resulting local edge feature,  $\phi''_{\omega,\theta}(x, y)$  at a pixel  $(x_c, y_c)$  is obtained as the absolute maximum of the features extracted with the family of generated SGWs kernels. More formally, let  $n_\omega$  be the number of scales used, and  $n_\theta$  the number of orientations, the resulting edge feature for a pixel  $(x_c, y_c)$  is calculated as:

$$\phi''_{\omega,\theta}(x_c, y_c) = \max \left\{ \phi'_{\omega_i,\theta_j}(x_c, y_c), i = 0, 1, \dots, n_\omega \text{ and } j = 0, 1, \dots, n_\theta \right\}. \quad (3.24)$$

In our experiments, we used two different scales  $\omega_0 = 0.3\pi$  and  $\omega_1 = 0.5\pi$ , and four

different orientations  $\theta_j = j\pi/4$  for  $j = 0, \dots, 3$  (JIANG; LAM; SHEN, 2009). Since edges are localized in an image, the size of the SGWs are used as either  $3 \times 3$  or  $5 \times 5$ . Two quantization levels for the positive and negative magnitudes of the GWs were used to create the SGWs (see Eq. (3.22) and Fig. 3.6), denoted  $q_1$  and  $q_2$ . Suppose a SGW is centered at the pixel position  $(x_c, y_c)$ , the resulting edge features for this SGW,  $\phi'_{\omega, \theta}(x_c, y_c)$  are computed efficiently as follows:

1.  $\omega = 0.3\pi$  and  $\theta = 0$  (Fig. 3.6 (a))

$$\begin{aligned} \phi'_{0,0}(x_c, y_c) = & q_1(I(x_c - 1, y_c + 1) + I(x_c + 1, y_c + 1) \\ & - I(x_c - 1, y_c - 1) - I(x_c + 1, y_c - 1)) \\ & + q_2((I(x_c, y_c + 1) - I(x_c, y_c - 1))), \end{aligned} \quad (3.25)$$

2.  $\omega = 0.3\pi$  and  $\theta = \pi/4$  (Fig. 3.6 (b))

$$\begin{aligned} \phi'_{0,1}(x_c, y_c) = & q_1(I(x_c, y_c + 2) + I(x_c + 2, y_c) \\ & - I(x_c, y_c - 2) - I(x_c - 2, y_c)) \\ & + q_2(I(x_c, y_c + 1) + I(x_c + 1, y_c) \\ & + I(x_c + 1, y_c + 1) - I(x_c, y_c - 1) \\ & + I(x_c - 1, y_c) - I(x_c - 1, y_c - 1)), \end{aligned} \quad (3.26)$$

3.  $\omega = 0.3\pi$  and  $\theta = \pi/2$  (Fig. 3.6 (c))

$$\begin{aligned} \phi'_{0,2}(x_c, y_c) = & q_1(I(x_c + 1, y_c - 1) + I(x_c + 1, y_c + 1) \\ & - I(x_c - 1, y_c - 1) - I(x_c - 1, y_c + 1)) \\ & + q_2(I(x_c + 1, y_c) - I(x_c - 1, y_c)), \end{aligned} \quad (3.27)$$

4.  $\omega = 0.3\pi$  and  $\theta = 3\pi/4$  (Fig. 3.6 (d))

$$\begin{aligned} \phi'_{0,3}(x_c, y_c) = & q_1(I(x_c, y_c - 2) \\ & + I(x_c + 2, y_c) - I(x_c - 2, y_c) - I(x_c, y_c + 2)) \\ & + q_2(I(x_c, y_c - 1) + I(x_c + 1, y_c) + I(x_c + 1, y_c - 1) \\ & - I(x_c - 1, y_c) - I(x_c - 1, y_c + 1) - I(x_c, y_c + 1)), \end{aligned} \quad (3.28)$$

5.  $\omega = 0.5\pi$  and  $\theta = 0$  (Fig. 3.6 (e))

$$\phi'_{1,0}(x_c, y_c) = q_2(I(x_c, y_c + 1) - I(x_c, y_c - 1)), \quad (3.29)$$

6.  $\omega = 0.5\pi$  and  $\theta = \pi/4$  (Fig. 3.6 (f))

$$\begin{aligned} \phi'_{1,1}(x_c, y_c) = q_2(I(x_c, y_c + 1) + I(x_c + 1, y_c) \\ - I(x_c, y_c - 1) - I(x_c - 1, y_c)), \end{aligned} \quad (3.30)$$

7.  $\omega = 0.5\pi$  and  $\theta = \pi/2$  (Fig. 3.6 (g))

$$\phi'_{1,2}(x_c, y_c) = q_2(I(x_c + 1, y_c) - I(x_c - 1, y_c)), \quad (3.31)$$

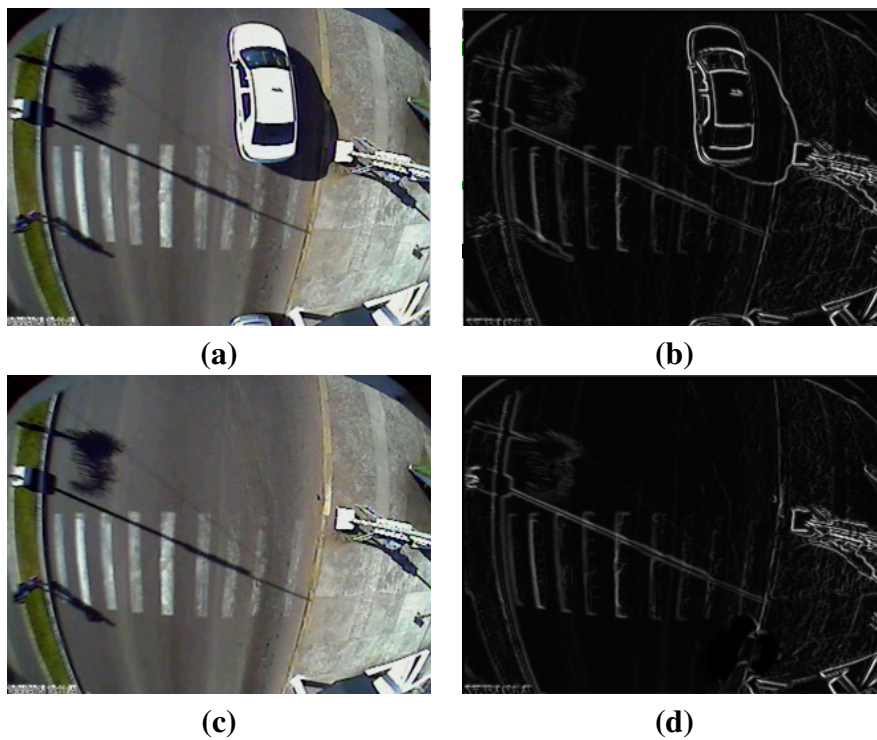
8.  $\omega = 0.5\pi$  and  $\theta = 3\pi/4$  (Fig. 3.6 (h))

$$\begin{aligned} \phi'_{1,3}(x_c, y_c) = q_2(I(x_c, y_c - 1) + I(x_c + 1, y_c) \\ - I(x_c, y_c + 1) - I(x_c - 1, y_c)). \end{aligned} \quad (3.32)$$

The computation of  $\phi'_{\omega,\theta}(x_c, y_c)$  requires no more than 2 multiplications and 22 additions (JIANG; LAM; SHEN, 2009), therefore the computational costs required to calculate the SGWs features at a given pixel  $(x_c, y_c)$  is much lower than by using GWs. This still can be improved if instead of calculating all the four orientations, we use only 2 orientations for each scale to determine if a pixel is an edge pixel, since the number of edge pixels usually is much smaller than the number of non-edge pixels (JIANG; LAM; SHEN, 2009). This is done by selecting the two SGWs with orientations  $\theta = 0$  and  $\theta = \pi/2$ , and scale  $\omega = 0.3\pi$  (*i.e.*  $\phi'_{0,0}(x, y)$  and  $\phi'_{0,2}(x, y)$ ). If a pixel  $(x_c, y_c)$  is an edge pixel, then either  $\phi'_{0,0}(x, y)$  or  $\phi'_{0,2}(x, y)$  should be larger than a threshold  $T_1$  (*i.e.* if both  $|\phi'_{0,0}(x, y)| < T_1$  and  $|\phi'_{0,2}(x, y)| < T_1$ , then the pixel  $(x_c, y_c)$  is not an edge pixel). If this is the case, the remaining SGW features  $\phi'_{\omega,\theta}$  do not need to be calculated, greatly reducing the computational cost (JIANG; LAM; SHEN, 2009). If the pixel  $(x_c, y_c)$  is a strong edge, then  $|\phi'_{0,0}(x, y) + \phi'_{0,2}(x, y)|$  will be larger than another threshold,  $T_2$ , and no further computation is needed. Otherwise, the other six  $\phi'_{\omega,\theta}$  are calculated.

At the end of this process, we obtain the edges for the frame  $F$  and background model  $B$ ,  $F_e$  and  $B_e$ , respectively, Fig. 3.7 illustrates the resulting edges calculated with the SGWs. The next step of our proposed method consists of using edge information and color information to validate the activation of the virtual loop  $L_i$ , determining if it in fact contains a vehicle.

Figure 3.7: Example of the edge detection using the SGW kernels: **(a)** original frame; **(b)** resulting frame edges,  $F_e$ ; **(c)** background model; and **(d)** resulting background edges,  $B_e$ .



Source: Created by the author (2015).

### 3.4 Vehicle Detection Using Edge and Color Information

The GMM foreground detection has the disadvantage of not being robust to sudden illumination changes, which causes false positives and leads to erroneous foreground detection results (BENEZETH et al., 2010). Pourmohammad, Fallahpour and Karimifar (2012) proposed using edge features and the NCC (Normalized Cross-Correlation) measure to perform template matching. This approach has the advantages of being robust to illumination changes and noise, and having low computational cost, allowing real-time applications (POURMOHAMMAD; FALLAHPOUR; KARIMIFAR, 2012). We adapted this template matching approach for our vehicle detection scheme using the edge information obtained as described in Section 3.3 to determine if a vehicle is in the virtual loop region  $L_i$ .

The Cross-Correlation and Normalized Cross-Correlation are well-known similarity measures and are widely used in template matching, tracking and stereo matching problems (POURMOHAMMAD; FALLAHPOUR; KARIMIFAR, 2012; HIRSCHMULLER; SCHARSTEIN, 2007; SEBASTIAN, 2007; HEO; LEE; LEE, 2011). Template matching consists in finding a given template in an image, and often this is done by sliding the template in the image and measuring a similarity score at each position. The use of cross-correlation for template matching is based on the squared Euclidean distance (LEWIS, 1995):

$$d_{f,t}^2(u, v) = \sum_{x,y} [f(x, y) - \mathcal{T}(x - u, y - v)]^2, \quad (3.33)$$

where  $f$  is the frame (image) and the sum is over all positions  $x, y$  in the window containing the template  $t$  positioned at  $(u, v)$ . By expanding Eq. (3.33) we obtain:

$$d_{f,t}^2(u, v) = \sum_{x,y} [f^2(x, y) - 2f(x, y)t(x - u, y - v) + \mathcal{T}^2(x - u, y - v)], \quad (3.34)$$

where the term  $\sum \mathcal{T}^2(x - u, y - v)$  is constant. Since the term  $\sum f^2(x, y)$  is approximately constant for a given frame, then the cross-correlation term representing the similarity measure between the frame and the template is defined as (LEWIS, 1995):

$$c(u, v) = \sum_{x,y} f(x, y)\mathcal{T}(x - u, y - v). \quad (3.35)$$

However, the approach described by Eq. (3.33) has a few limitations, such as it is not invariant to illumination changes across the scene, the range of  $c(u, v)$  is dependent of the size

of the template, and if the image energy  $\sum f^2(x, y)$  varies with location  $(u, v)$ , the correlation between the template and the exact matching region may result in a lower similarity measure value than between the template and an image bright spot (LEWIS, 1995). Therefore, to overcome these limitations the image and template vectors are normalized to unit length, obtaining the Normalized Cross-Correlation measure as follows (BRIECHLE; HANEBECK, 2001):

$$\gamma = \frac{\sum_{x,y} [f(x, y) - \bar{f}_{u,v}] [\mathcal{T}(x - u, y - v) - \bar{\mathcal{T}}]}{\sqrt{\sum_{x,y} [f(x, y) - \bar{f}_{u,v}]^2 \sum_{x,y} [\mathcal{T}(x - u, y - v) - \bar{\mathcal{T}}]^2}}, \quad (3.36)$$

where  $f(x, y)$  is the intensity value of the image  $f$  of size  $M_x \times M_y$  at the pixel  $(x, y)$ ,  $x \in \{0, \dots, M_x - 1\}$ ,  $y \in \{0, \dots, M_y - 1\}$ ,  $\mathcal{T}$  is the template of size  $N_x \times N_y$ ,  $\bar{f}$  and  $\bar{\mathcal{T}}$  are the mean intensity values of the window of size  $N_x \times N_y$  centered at the position  $(u, v)$  of the template  $\mathcal{T}$  on the image  $f$ , respectively. The NCC measure  $\gamma \in [0, 1]$ , and a score of NCC=1 indicates an exact match. In template matching, the NCC value  $\gamma$  for a given template  $\mathcal{T}$  is evaluated at each point  $(u, v)$  of the image  $f$ , such that  $u$  and  $v$  defines the shift in the  $x$  and  $y$  directions, respectively, of the template  $\mathcal{T}$ . The position  $(u, v)$  that results in the highest  $\gamma$  value is considered to be the correct match.

In our vehicle detection scheme, we want to measure the NCC similarity between the frame edges  $F_e$  and background edges  $B_e$  in the virtual loop region  $L_i$ . Therefore, we can rewrite Eq. (3.36) as follows:

$$\gamma^{L_i} = \frac{\sum_{x,y} [B_e^{L_i}(x, y) - \bar{B}_e^{L_i}] [F_e^{L_i}(x, y) - \bar{F}_e^{L_i}]}{\sqrt{\sum_{x,y} [B_e^{L_i}(x, y) - \bar{B}_e^{L_i}]^2 \sum_{x,y} [F_e^{L_i}(x, y) - \bar{F}_e^{L_i}]^2}}, \quad (3.37)$$

where  $F_e^{L_i}$  and  $B_e^{L_i}$  are the edges magnitudes of the frame  $F$  and background  $B$  (see Section 3.3) in the virtual loop region  $L_i$ , and  $\bar{F}_e^{L_i}$  and  $\bar{B}_e^{L_i}$  are the mean values of the edges magnitudes of the frame  $F$  and background  $B$  in the virtual loop region  $L_i$ . If the edges magnitudes  $F_e^{L_i}$  and  $B_e^{L_i}$  are similar (*i.e.* high  $\gamma^{L_i}$  value), we consider that this region does not contain any vehicle and  $\mathcal{V}_i^t = 0$ . Otherwise, if the difference is significant (*i.e.* low  $\gamma^{L_i}$  value), it is assumed that there is a vehicle in the region  $L_i$  at the time instant  $t$ , and we set  $\mathcal{V}_i^t = 1$ .

Since vehicles usually are longer than the virtual loop regions  $L_i$ , using only edge information tends to provide inaccurate vehicle detection results, especially when only the vehicles smooth regions are inside the region (*i.e.* there is not enough edge information in the vehicle region). To overcome this limitation and improve our detection results, we also compare color histograms using the Bhattacharyya distance (ZWENG; RITTLER; KAMPEL, 2011), with the

color histogram similarity calculated as follows:

$$d_H(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{(H_1(i) \cdot H_2(i))}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}}, \quad (3.38)$$

where  $H_1$  and  $H_2$  are the two histograms, and  $i$  represents the color bins. The Bhattacharyya distance evaluates to '0' if the histograms match completely. The last step of our proposed vehicle detection method is to combine edge and the color information to determine the final value of  $\mathcal{V}_i^t$ .

First, we define two NCC thresholds,  $\tau_\gamma^{\text{low}} \in [0, 1]$  and  $\tau_\gamma^{\text{high}} \in (\tau_\gamma^{\text{low}}, 1]$ . If for a virtual loop  $L_i$ ,  $\gamma^{L_i} \leq \tau_\gamma^{\text{low}}$  (see Eq. (3.37)), then the difference is considered significant and a vehicle is assumed to be in the virtual loop region  $L_i$  (i.e.  $\mathcal{V}_i^t = 1$ ). If  $\gamma^{L_i} > \tau_\gamma^{\text{high}}$  then a high similarity is assumed between the frame edges  $F_e$  and the background edges  $B_e$  in the virtual loop region  $L_i$ , and no vehicle is detected in  $L_i$  (i.e.  $\mathcal{V}_i^t = 0$ ). However, for the case  $\tau_\gamma^{\text{low}} < \gamma^{L_i} \leq \tau_\gamma^{\text{high}}$ , if the virtual loop  $L_i$  has been active for at least  $\tau_l$  frames, we compare the histogram of the current frame  $F^t$  in the virtual loop region  $L_i$  with the histograms of the previous  $l$  frames in this region with  $\mathcal{V}_i^k = 1$ ,  $k = \{t-l, \dots, t-1\}$ , where  $l$  is the number of previous frames in which a vehicle in the virtual loop region  $L_i$  was detected. The motivation to include this step is the following: If a virtual loop  $L_i$  was active before (i.e.  $\mathcal{V}_i^t = 1$ ) when a vehicle was detected in the region  $L_i$ , it should only be deactivated (i.e.  $\mathcal{V}_i^t = 0$ ) when the vehicle leaves the region  $L_i$  completely. Therefore, if the histogram of the current frame  $F^t$  in  $L_i$  is similar to the histograms of the previous frames in  $L_i$ , then we can assume that the vehicle is still in the virtual loop  $L_i$ . In our experiments,  $\tau_\gamma^{\text{low}}$  and  $\tau_\gamma^{\text{high}}$  were adjusted for each video sequence to obtain the best results (see Section 4.2).

To describe this process more formally, let  $\mathcal{F} = \{F_{L_i}^l, \dots, F_{L_i}^{t-1}\}$  be the set of  $l$  previous frames that showed a vehicle in the virtual loop region  $L_i$  (i.e.  $\mathcal{V}_i^t = 1$ ). We proceed to the next step if the virtual loop  $L_i$  has been active for at least  $\tau_l$  frames as follows:

$$C_i^t = \sum_{k=t-l-1}^{t-1} \delta(\mathcal{V}_i^k, 1) \geq \tau_l, \quad (3.39)$$

where  $\delta(a, b)$  is the Kronecker delta such that  $\delta(a, b) = 1$  if  $a = b$ , and  $\delta(a, b) = 0$  otherwise. If the condition in Eq. (3.39) is satisfied, we measure the similarity of the histogram of the virtual loop region  $L_i$  of the current frame  $F_i^t$  with each one of the histograms of  $L_i$  of the previous active frames, and estimate how many of the previous frames have similar  $L_i$  histograms, and



this is expressed as follows:

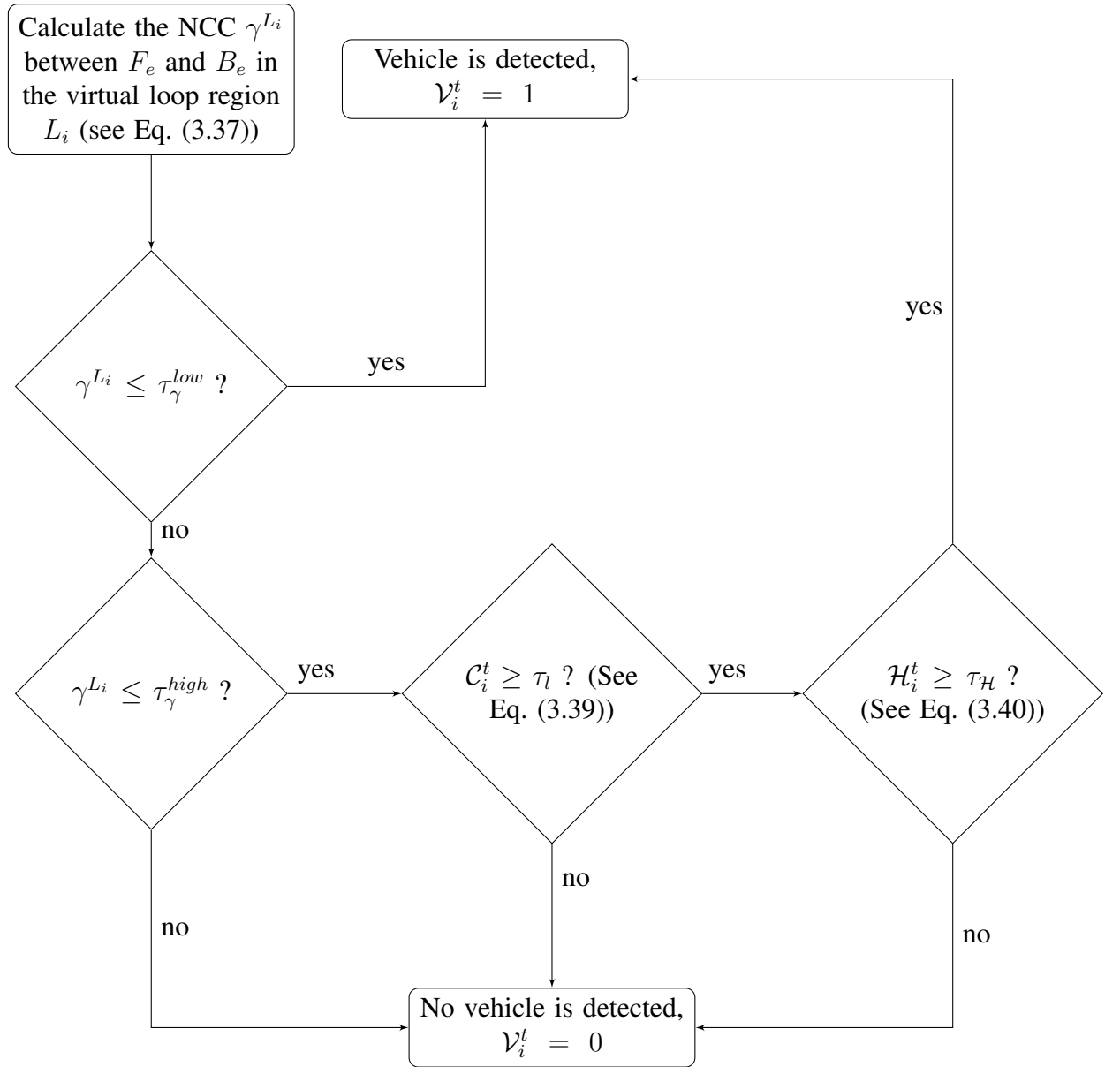
$$\mathcal{H}_i^t = \frac{\sum_{F^k \in \mathcal{F}} u(\tau_u - d_H(H_{F_i^t}, H_{F_i^k}))}{l}, \quad (3.40)$$

where  $u(\cdot)$  is the unit step function, which evaluates to ‘1’ if  $d_H(H_{F_i^t}, H_{F_i^k}) \leq \tau_u \in [0, 1]$ , or to ‘0’ otherwise;  $d_H(\cdot)$  is the histogram similarity function and is given by the Bhattacharyya distance (see Eq. (3.38));  $H_{F_i^k}$  is the histogram of the  $k$ -th frame, and  $H_{F_i^t}$  is the histogram of the current frame in the virtual loop region  $L_i$ , and  $l$  is the number of elements in the set  $\mathcal{F}$ . In our experiments, the value of  $\tau_u$  was adjusted experimentally for each video sequence (see Section 4.2). Since vehicles usually are longer than virtual loops, by using the temporal detection history of the virtual loop region false negatives can be avoided (e.g. caused by smooth regions of the vehicles with low edge content), such that the virtual loop reports ‘0’ only after the vehicle left the loop  $L_i$  completely. Therefore,  $\mathcal{H}_i^t$  evaluates the similarity ratio between the active loop  $L_i$  in the current frame and the active loop  $L_i$  in previous frames. If this ratio is higher than a threshold  $\tau_{\mathcal{H}}$ , we can assume that a vehicle is still passing through this loop region  $L_i$ . The final detection result for a virtual loop  $L_i$  is defined as follows:

$$\mathcal{V}_i^t = \begin{cases} 1, \gamma^{L_i} \leq \tau_{\gamma}^{low} \text{ or,} \\ (\gamma^{L_i} \leq \tau_{\gamma}^{high} \text{ and } C_i^t \geq \tau_l \text{ and } \mathcal{H}_i^t \geq \tau_{\mathcal{H}}), \\ 0, \text{ otherwise.} \end{cases} \quad (3.41)$$

Fig. 3.8 summarizes this vehicle detection process.

Figure 3.8: Schematics of the vehicle detection process using both edge and color information (see Eq. (3.41)).

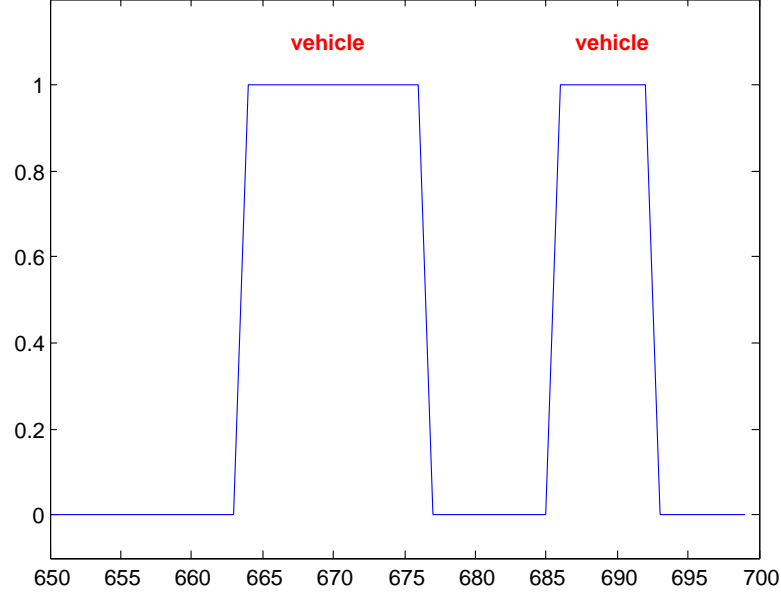


Source: Created by the author (2015).

### 3.4.1 Vehicle Counting

We can use the measured vehicle detection results to count the number of vehicles that passed in a virtual loop region  $L_i$ . This is performed by analyzing the variation in the binary signal obtained by the vehicle detection result. A vehicle in a virtual loop  $L_i$  is counted when the signal varies from '0' to '1' and then back to '0'. Fig. 3.9 illustrates vehicles binary signal in a virtual loop  $L_i$ . To prevent noise, only vehicles that activated the virtual loop  $L_i$  for at least  $\tau_v$

Figure 3.9: Example of the binary signal representing passing vehicles in a virtual loop  $L_i$  for a few frames of one of our video sequences. Each peak represents a passing vehicle.



Source: Created by the author (2015).

frames are counted. To describe it more formally: Let  $Z_i = \{\mathcal{V}_i^{t-l}, \dots, \mathcal{V}_i^t\}$  be the binary signal in the virtual loop  $L_i$  of the previous  $l$  frames, such that  $\mathcal{V}_i^{t-l-1} = 0$  and  $\mathcal{V}_i^{t-l} = 1$  (i.e. the virtual loop was inactive and became active). A vehicle is counted only if:

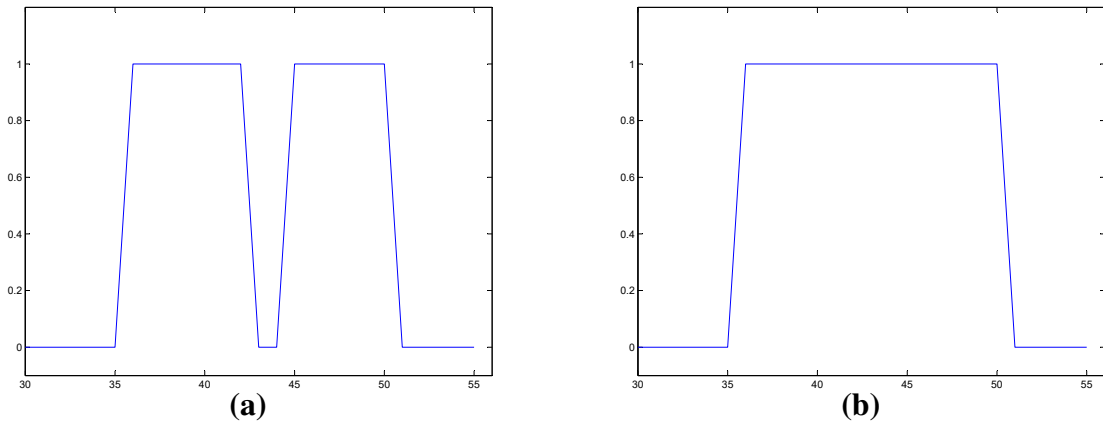
$$\begin{cases} \mathcal{V}_i^{t-1} = 1 \text{ and } \mathcal{V}_i^t = 0, \text{ and} \\ \sum_{\mathcal{V} \in Z_i} \mathcal{V} \geq \tau_v \end{cases} \quad (3.42)$$

where  $\mathcal{V}_i^{t-1} = 1$  and  $\mathcal{V}_i^t = 0$  is the vehicle detection result for the current frame and previous frame, and  $Z_i$  is the binary signal of the previous  $l$  frames. To reduce the effects of noise in the vehicle detection binary signal  $Z_i$ , a median filter with size  $w$  is applied. The median filter is chosen due to its simplicity and edge preserving properties. It reduces the number of false positives that could be caused by variations in the signal while one vehicle is passing the virtual loop  $L_i$ .

To prevent counting errors when a vehicle passes through adjacent virtual loops at the same time (i.e. when changing lanes), the vehicle detection signal in adjacent virtual loops regions  $L_i$  and  $L_{i+1}$  (i.e.  $Z_i$  and  $Z_{i+1}$ ) is compared according to the following equation:

$$d(Z_i, Z_{i+1}) = 1 - \frac{\sum_{\mathcal{V}_a \in Z_i, \mathcal{V}_b \in Z_{i+1}} |\mathcal{V}_a - \mathcal{V}_b|}{l} \quad (3.43)$$

Figure 3.10: Example of median filter applied to a binary signal  $Z_i$  **(a)** binary signal with noise; **(b)** filtered binary signal with a median filter.



Source: Created by the author (2015).

where  $d(Z_i, Z_{i+1})$  is the signal similarity such that  $d(Z_i, Z_{i+1}) = 1$  means that the signals are identical and  $l$  is the number of elements in the sets  $Z_i$  and  $Z_{i+1}$ .

If the signals are sufficient similar (*i.e.*  $d(Z_i, Z_{i+1}) \geq \tau_z$ ), we assume they belong to the same vehicle and use the foreground mask  $F'_G$  to calculate the number of foreground pixels accumulated in the regions  $L_i$  and  $L_{i+1}$  during the vehicle passage. Finally, the vehicle count is incremented only for the region that accumulated the highest number of foreground pixels. The values of  $\tau_v$  and  $\tau_z$  were adjusted experimentally for each video sequence (see Section 4.2).

In the next chapter, we describe our experiments and results.







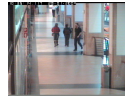
## 4 EXPERIMENTAL RESULTS

In this chapter, we present our shadow detection and vehicle detection and counting results, describe our experimental setup and discuss implementation details. First, we present our hypergraph-based shadow detection approach results, followed by our vehicle detection scheme results.

### 4.1 Hypergraph-Based Shadow Detection Results

In order to validate our hypergraph-based shadow detection scheme, we performed experiments on seven popular video sequences that are publicly available (SANIN et al., 2012), and measured the shadow detection ( $\eta$ ) and shadow discrimination ( $\xi$ ) rates. We also used the average of these two measures ( $\frac{\eta+\xi}{2}$ ) as a third measure (SANIN et al., 2012). We chose those video sequences because they are commonly used to validate shadow detection methods (SANIN et al., 2012), and have the ground truth data and foreground masks available. Summarizing information about these datasets can be seen in Table 4.1.

Table 4.1: Information about the datasets used in the experiments.

Dataset	Highway 1	Lab	Highway 3	Campus	Room	Hallway	CAVIAR
Frame Example							
Number of frames	8	14	7	53	22	13	1125
Resolution	320x245	320x240	320x240	320x240	320x245	320x240	384x288
Scene	Outdoor	Indoor	Outdoor	Outdoor	Indoor	Indoor	Indoor
Objects	Vehicles	People	Vehicles	People	People	People	People
Shadow	Strong	Light	Strong	Light	Light	Light	Light

Source: created by the author (2015).

For the comparative methods, we chose the methods in (HUANG; CHEN, 2009) (SANIN et al., 2012) (BULLKICH et al., 2012).

The method proposed by Huang and Chen (2009) uses physically-based principles, and tries to learn the appearance of shadow pixels. Sanin et al. (2012) proposed to use chromaticity and gradient information to achieve high shadow detection ( $\eta$ ) and discrimination ( $\xi$ ) rates. The method proposed by Bullkich et al. (2012) uses the MTM (Matching by Tone Mapping) distance between image patches to detect shadows. For more details of the comparative methods see Section 2.3.

We implemented our shadow detection scheme using MATLAB and used the hMETIS (hMETIS..., ) hypergraph partitioning library. In our experiments, parameters were adjusted experimentally. For the image hypergraph construction, we set the grid distance  $\beta = 2$ . For sequences with strong shadows, the color distance  $\lambda = 10$ . For videos with light shadows, the color distance of  $\lambda = 4$  produced more accurate results (see Eq. (A.4)). Also, we used the number of partitions  $K = 51$ . The parameters we used for the hMETIS tool are summarized in Table 4.2.

The sequences Highway 1 and Highway 3 contain vehicles on a road and their cast shadows. Some passing vehicles have dark colors, similar to their shadows, generating difficulties for chromaticity-based shadow detection methods to discriminate between the vehicles and their shadows.

The method proposed in (SANIN; SANDERSON; LOVELL, 2010) combines color and gradient information and obtains better results, but it requires large color differences between shadows and vehicles to perform well. Our stochastic hypergraph-based shadow detection scheme combines color and gradient information, as well as information on the pixels vicinities, to classify the segmented image regions as shadow or non-shadow, and that may explain its improved shadow detection and shadow/vehicle discrimination rates (see Table 4.3).

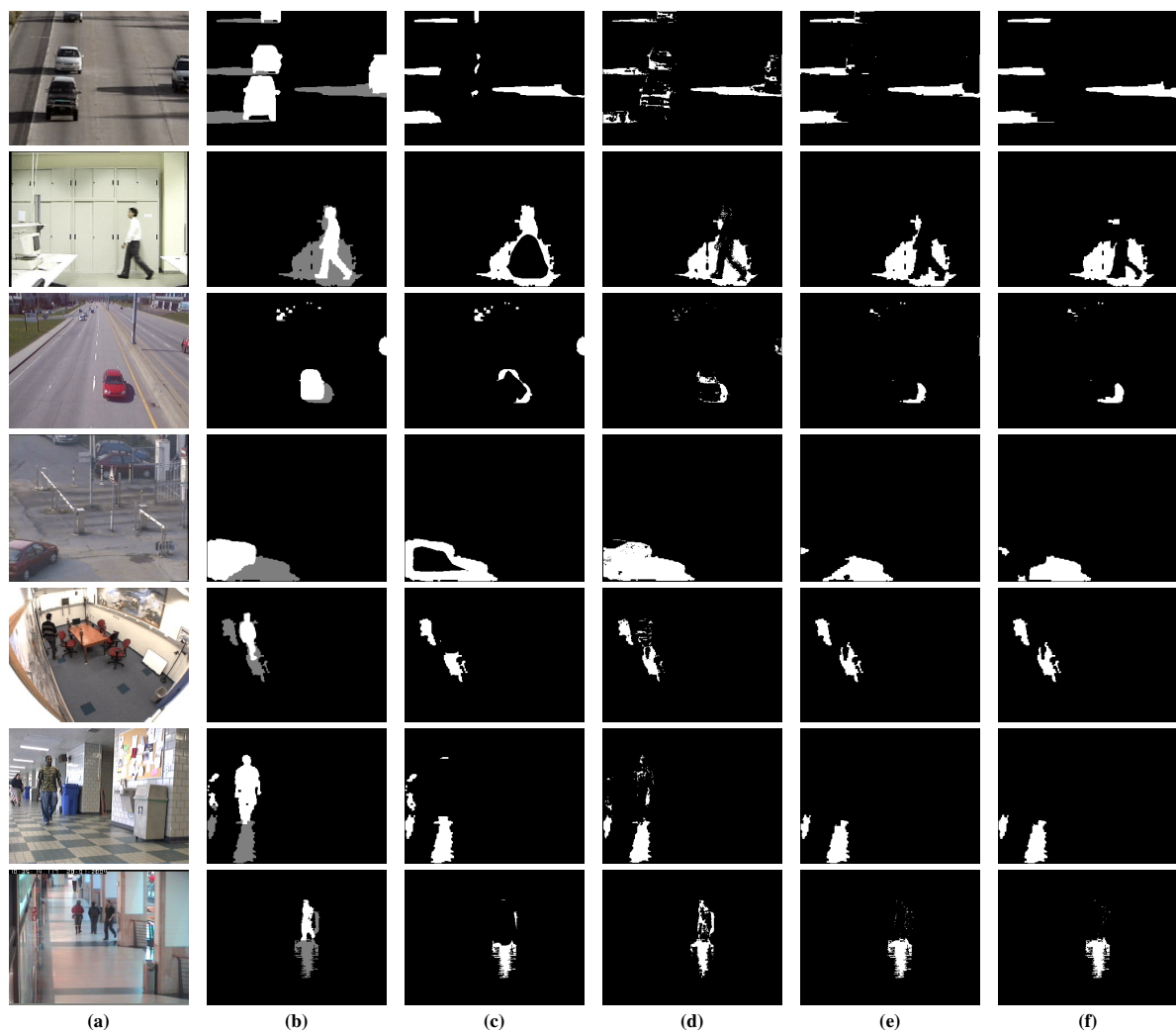
The video sequences Lab, Campus, Hallway, Room and CAVIAR contain light shadows cast by people walking in different environments. Despite the shadows being light, the foreground objects and shadows do not have similar colors, avoiding the chromatic problems mentioned before and improving the shadow detection results for all methods (see Table 4.3).

To illustrate the variability occurring in the shadow masks  $\hat{\mathcal{X}}_p(i, j)$  due to the stochasticity of the hypergraph partition algorithm, we measured the worst and best results of 10 different hypergraph partitionments obtained in our tests. We also measured the results obtained using the stochastic majority voting scheme expressed by Eq. (3.18) for  $N = 10$  and  $\tau_r = 3$  (see Table 4.3). Qualitative results obtained by our proposed method and the comparative methods can be seen in Fig. 4.1. Visually, the shadow masks generated by our method tend to be more accurate than those obtained by comparative methods, representing the state of the art in shadow detection (HUANG; CHEN, 2009) (SANIN et al., 2012) (BULLKICH et al., 2012).

Quantitative results of the tests with our proposed method and for the comparative methods (HUANG; CHEN, 2009) (SANIN et al., 2012) (BULLKICH et al., 2012) for each dataset, and the average results considering all the seven sequences can be seen in Table 4.3.

For the seven video sequences our method with the proposed voting scheme (see Eq. (3.18)) obtained better detection rates ( $\eta$ ) than all comparative methods and comparable discrimination

Figure 4.1: Shadow masks obtained by the proposed hypergraph-based shadow detection scheme and by the comparative methods for the datasets. **(a)** original frame; **(b)** ground-truth mask; **(c)** shadow mask obtained by the method proposed by Bullick et al. (2012); **(d)** shadow mask obtained by the method proposed by Huang and Chen (2009); **(e)** shadow mask obtained by the method proposed by Sanin, Sanderson and Lovell (2010); **(f)** shadow mask obtained by the method proposed by our proposed method.



Source: Created by the author (2015).

Table 4.2: Parameters used to set the hMETIS tool package for our tests.

Parameter	Value
Number of partitions	51
Number of runs	10
UB factor	1
Matching Scheme	Hyperedge
Refinement Policy	Fiduccia-Mattheyses(FM)
V-cycle Refinement	On good intermediate solution
Reconst	Reconstructs
Color distance ( $\lambda$ )	10 (strong shadows) and 4 (light shadows)
Grid distance ( $\beta$ )	2

Source: Created by the author (2015).

rates (see Table 4.3), resulting in a higher average measure  $\frac{\eta+\xi}{2}$  for all video sequences, except in the CAVIAR sequence for which the method proposed by Bullkich et al. (2012) obtained a higher average measure. Even by considering the worst results of the 10 executions, our method still performed better than the comparative methods for most sequences. By averaging the results for all video sequences, our method obtained a better detection rate ( $\eta$ ) than all comparative methods in the worst case, best case and by using the proposed voting scheme (see Eq. (3.18)). The obtained average discrimination rate  $\xi$  was higher than the average discrimination rates obtained by (BULLKICH et al., 2012) and (HUANG; CHEN, 2009), and comparable to the  $\xi$  obtained by (SANIN; SANDERSON; LOVELL, 2010). In average, our shadow detection results using the  $\frac{\eta+\xi}{2}$  measure was better than those obtained by all comparative methods.

The experimental evidence suggests that our proposed method with the stochastic majority voting rule, potentially helps obtaining more accurate shadow detection in video sequences. Our results show that the proposed shadow detection scheme potentially can achieve better qualitative and quantitative results than recent shadow detection methods, that are representative of the state-of-the-art in shadow detection. However, our current implementation was unable to perform in real-time, therefore we decided not to use this approach in the vehicle detection scheme.

In the next section, we present our vehicle detection results.



Figure 4.2: Example of the changes in camera gain caused by the buses passing through the virtual loop too close to the camera (top view).



Source: Created by the author (2015).

## 4.2 Vehicle Detection Results

In order to evaluate our vehicle detection scheme, and compare it to other approaches representative of the state of the art, we performed experiments on 3 challenging urban traffic video sequences with top views. The virtual loops  $L_i$  were defined manually, and there is one virtual loop per lane. The first video sequence contains light cast shadows and buses that change the camera gain as they pass through the virtual loops, which translates into a sudden change in illumination (see Fig. 4.2). The second video sequence does not contain shadows, but contains buses passing through the virtual loops, causing changes in the camera gain (illumination sudden changes). The third video sequence contains vehicles with their cast shadows. Table 4.4 contains summarizing data about the 3 video sequences.

We implemented our method in C++ and OpenCV. The simplified Gabor edges detector was implemented in GPU using OpenCL to improve the efficiency. For the methods in (ZIVKOVIC; HEIJDEN, 2006), (NOH; JEON, 2013) and (ST-CHARLES; BILODEAU, 2014) the background subtraction library in (SOBRAL, 2013) was used. Our experiments were performed in a computer with an Intel Core I5 M520 2.40 Ghz CPU, 8 GB ram and an Ati Mobility HD 5650 video card running Windows 10 operating system.

To evaluate the performance of our method, we used several measures: accuracy, detection error, predictive value (or precision), sensitivity (or recall) and f-score. These measures are defined as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad (4.1)$$

$$\text{Detection Error} = \frac{FP + FN}{TP + TN + FP + FN}, \quad (4.2)$$

$$\text{Predictive value} = \frac{TP}{TP + FP}, \quad (4.3)$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \quad (4.4)$$

$$\text{F-Score} = 2 \times \frac{\text{Predictive value} \times \text{Sensitivity}}{\text{Predictive value} + \text{Sensitivity}}, \quad (4.5)$$

where  $TP$  are the true positives,  $FP$  are the false positives,  $TN$  are the true negatives and  $FN$  are the false negatives. A true positive occurs when a vehicle was correctly detected at the virtual loop (*i.e.*  $\mathcal{V}_i^t = GT_{L_i}^t = 1$ ), where  $GT_{L_i}^t$  is the ground-truth value for the virtual loop  $L_i$  at the time instant  $t$ . A false positive occurs a vehicle is detected in the virtual loop  $L_i$  (*i.e.*  $\mathcal{V}_i^t \neq GT_{L_i}^t$ ) and  $GT_{L_i}^t = 0$ . A true negative occurs when the sensor correctly detects that no vehicle was in the virtual loop  $L_i$  (*i.e.*  $\mathcal{V}_i^t = GT_{L_i}^t = 0$ ). A false negative occurs when the sensor does not detect a vehicle present in the virtual loop  $L_i$  (*i.e.*  $\mathcal{V}_i^t \neq GT_{L_i}^t$  and  $GT_{L_i}^t = 1$ ). The ground-truths were obtained manually, and for each frame we verified if vehicles were inside the virtual loop regions. We measured the values for each lane to obtain the detection results for a video sequence, then we considered all the lanes as a single lane and applied the measures described before (*i.e.*  $TP = TP_1 + TP_2 + TP_3$ ,  $FP = FP_1 + FP_2 + FP_3$ ,  $TN = TN_1 + TN_2 + TN_3$  and  $FN = FN_1 + FN_2 + FN_3$ ). The accuracy measures all the times the system correctly detected (or not detected) a vehicle in  $L_i$ . Predictive value and sensitivity measures only consider the time instants when a vehicle is in the virtual loop  $L_i$ . The predictive value decreases when the system incorrectly detects a vehicle in  $L_i$ , and the sensitivity decreases when the system incorrectly detects that no vehicle is in  $L_i$ . There is a compromise between the predictive value and the sensitivity, therefore the F-Score tries to combine them into a single measure. The results for each video sequence and the processing times for each method are presented in Tables 4.5 to 4.7.

To the best of our knowledge, no other method that simulates a virtual inductive loop is available in the literature. Therefore, we chose recent and relevant foreground detection schemes to compare with our proposed method. We compared our proposed vehicle detection scheme with the methods in (ZIVKOVIC; HEIJDEN, 2006), (NOH; JEON, 2013) and (ST-CHARLES; BILODEAU, 2014). By measuring the accuracy, detection error, predictive value and sensitiv-

ity as well as the processing time per frame. Results for the 3 video sequences can be seen in Table 4.5, Table 4.6 and Table 4.7.

The first video sequence contains light cast shadows and sudden camera gain changes, but the proposed method showed robustness to these situations in our experiments (see Table 4.5). The method proposed by (ZIVKOVIC; HEIJDEN, 2006) was not affected by the light shadows, but was not robust against the sudden illumination changes. The method proposed by (NOH; JEON, 2013) was not affected by the light shadows, but presented detection errors against the sudden illumination changes. The method proposed by (ST-CHARLES; BILODEAU, 2014) presented the best results among the comparative methods obtaining accuracies higher than 90%, but it was not robust against sudden illumination changes. Our proposed method was able to achieve more accurate results than the comparative method with a low computational cost.

The second video sequence does not contain shadows but contains sudden illumination changes. The comparative methods had slightly worse results for the second video sequence, due to more illumination changes situations (see Table 4.6). The proposed method obtained the most accurate results and was not affected by illumination changes.

The third video sequence contains strong cast shadows that can really affect negatively the results. The proposed method was relatively robust to the shadows, and was able to obtain an average of 98.48% accuracy, but shadows tend to affect negatively the predictive value and the sensitivity. The comparative methods were not robust against the shadows, obtaining lower predictive value (i.e. higher false positives) and F-Score measures. The proposed method was able to obtain more accurate results, with an average of 98.48% accuracy and 89.6% F-Score measure (see Table 4.7). There is a compromise between the shadow detection rate and shadow discrimination rate (SANIN et al., 2012). By adjusting the parameters to increase the shadow detection rate, the false positives caused by shadows tends to decrease, leading to higher predictive values. However this would decrease the shadow discrimination rate, and the sensitivity measure is negatively affected. Therefore, we tuned the shadow detection parameters to achieve the highest F-Score measure. We adjusted experimentally the parameters for each video sequence, namely the  $\tau_{\gamma}^{\text{low}}$ ,  $\tau_{\gamma}^{\text{high}}$ ,  $\tau_l$  and  $\tau_{\mathcal{H}}$ . A summary of the parameters values used in our experiments can be seen on Table 4.8.








The proposed vehicle detection approach was able to achieve an average accuracy higher than 98% in all three video sequences. For the videos 1 and 2, high predictive value, sensitivity and F-Score also were obtained (higher than 96%). We also compared the processing time of all methods (see Table 4.5, Table 4.6 and Table 4.7). Our method had a processing time just a little higher than the method proposed in (ZIVKOVIC; HEIJDEN, 2006) and lower than the

comparative methods in (NOH; JEON, 2013) and (ST-CHARLES; BILODEAU, 2014). The highest processing time was for video sequence 3, due to the shadow detection and removal schemes. The proposed method potentially is suitable for real-time processing.

To evaluate our vehicle counting scheme, we compared with the method proposed by (YUAN; ZHAO; WANG, 2013). This method was chosen because it is a recent vehicle counting method that does not perform tracking. The counting ground-truths (including the false positives and false negatives) were obtained manually. The results can be seen on Tables 4.9 to 4.11. For the first video sequence, both the method in (YUAN; ZHAO; WANG, 2013) and our proposed method counted the total number of vehicles correctly, but our proposed method presented less false positives (vehicles counted multiple times) and false negatives (vehicles not counted). For the video sequence 2, our proposed method had a more accurate counting and also less false positives and false negatives. For video sequence 3 which contains strong cast shadows, the method proposed by (YUAN; ZHAO; WANG, 2013) resulted in a less accurate count (1.68% error) with more false positives and negatives (57 and 60, respectively). Our proposed method presented a more accurate total vehicle count (1.12% error), and less false positives and false negatives (33 and 31, respectively).




The experiments show that our vehicle detection scheme can potentially be used to accurately count vehicles in traffic video sequences with challenging situations. In the next chapter, we present our conclusions.

Table 4.3: Shadow detection results for all datasets and the average results. For each dataset, first row is the shadow detection rate  $\eta$ , second row is the shadow discrimination rate  $\xi$  and third row is the average of the two measures  $\frac{\eta+\xi}{2}$ .

Datasets	Methods					
	Bullkich et al. (2012)	Huang and Chen (2009)	Sanin et. al. (2010)	P. Worst	P. Best	P. Voting
	72.00%	50.49%	81.84%	86.75%	87.92%	87.89%
	95.08%	84.81%	94.26%	93.50%	94.67%	94.34%
	83.54%	67.65%	88.05%	90.12%	91.29%	91.11%
	67.94%	82.11%	85.99%	91.14%	91.13%	92.50%
	73.69%	84.96%	96.65%	93.85%	94.46%	93.23%
	70.82%	83.53%	91.32%	92.49%	92.80%	92.86%
	79.61%	43.88%	62.23%	67.62%	69.10%	64.94%
	60.88%	70.95%	90.68%	85.29%	86.78%	90.14%
	70.25%	57.42%	76.45%	76.46%	77.94%	77.54%
	74.59%	80.90%	77.74%	80.39%	82.25%	84.36%
	77.60%	48.22%	83.39%	80.66%	80.69%	78.95%
	76.10%	64.56%	80.56%	80.52%	81.47%	81.66%
	72.86%	80.72%	91.36%	91.44%	92.11%	92.47%
	97.48%	89.91%	94.50%	93.98%	93.91%	93.47%
	85.17%	85.31%	92.93%	92.71%	93.01%	92.97%
	90.43%	82.62%	95.18%	94.53%	94.71%	95.16%
	77.06%	87.73%	95.73%	95.92%	96.87%	95.94%
	83.74%	85.18%	95.46%	95.22%	95.79%	95.55%
	91.57%	84.52%	93.77%	93.66%	93.80%	94.76%
	86.76%	68.22%	82.07%	82.52%	82.62%	81.15%
	89.16%	76.37%	87.92%	88.09%	88.21%	87.96%
Average	78.43%	72.18%	83.82%	86.50%	87.28%	87.44%
	81.22%	76.40%	90.84%	89.38%	90.00%	89.60%
	79.83%	74.29%	87.33%	87.94%	88.64%	88.52%

Source: Created by the author (2015).

Table 4.4: Summarizing information about the videos used in the experiments.

	Video 1	Video 2	Video 3
<b>Dataset</b>			
Number of frames	4850	8976	17648
Resolution	$320 \times 240$	$320 \times 240$	$320 \times 240$
Shadows	Light cast shadows	No shadows	Strong cast shadows
Camera gain changes	Yes	Yes	No

Source: Created by the author (2015).

Video 1 results							
Method	Lane	Accuracy	D. Error	Predictive Value	Sensitivity	F-Score	Time per frame
<b>Zivkovic (2006)</b>	1	98.37%	1.63%	86.07%	93.77%	89.75%	33.49 ms
	2	98.27%	1.73%	80.72%	97.21%	88.20%	
	3	98.33%	1.67%	80.71%	96.74%	88.00%	
	Avg.	<b>98.32%</b>	<b>1.68%</b>	<b>82.57%</b>	<b>95.80%</b>	<b>88.69%</b>	
<b>Noh (2013)</b>	1	97.96%	2.04%	95.00%	77.24%	85.20%	107.4 ms
	2	99.24%	0.76%	97.04%	91.33%	94.10%	
	3	98.71%	1.29%	96.99%	83.88%	89.96%	
	Avg.	<b>98.71%</b>	<b>1.28%</b>	<b>96.99%</b>	<b>83.88%</b>	<b>89.96%</b>	
<b>St-Charles (2014)</b>	1	98.91%	1.09%	92.93%	92.68%	92.81%	276 ms
	2	99.26%	0.74%	92.84%	96.28%	94.53%	
	3	98.64%	1.36%	90.85%	87.30%	89.04%	
	Avg.	<b>98.93%</b>	<b>1.07%</b>	<b>92.28%</b>	<b>92.19%</b>	<b>92.24%</b>	
<b>Proposed</b>	1	99.26%	0.07%	96.37%	93.76%	95.05%	35.40 ms
	2	99.82%	0.02%	99.37%	97.83%	98.59%	
	3	99.50%	0.05%	93.53%	99.02%	96.20%	
	Avg.	<b>99.53%</b>	<b>0.05%</b>	<b>96.40%</b>	<b>95.80%</b>	<b>96.51%</b>	

Table 4.5: Vehicle detection results for video sequence 1.

Source: Created by the author (2015).

Table 4.6: Vehicle detection results for video sequence 2.

Video 2 results							
Method	Lane	Accuracy	Detection Error	Predictive Value	Sensitivity	F-Score	Time per frame
<b>Zivkovic (2006)</b>	1	96.95%	3.05%	85.16%	87.41%	86.27%	31.49 ms
	2	96.90%	3.10%	86.66%	90.72%	88.64%	
	3	95.05%	4.95%	60.17%	90.95%	72.42%	
	Avg.	<b>96.30%</b>	<b>3.70%</b>	<b>78.25%</b>	<b>89.62%</b>	<b>83.55%</b>	
<b>Noh (2013)</b>	1	97.13%	2.87%	99.06%	74.52%	85.05%	105.6 ms
	2	97.504%	2.50%	98.70%	82.36%	89.79%	
	3	99.29%	0.71%	98.33%	91.58%	94.83%	
	Avg.	<b>97.97%</b>	<b>2.03%</b>	<b>98.72%</b>	<b>81.72%</b>	<b>89.41%</b>	
<b>St-Charles (2014)</b>	1	98.34%	1.66%	96.14%	88.43%	92.12%	275.4 ms
	2	98.20%	1.80%	95.03%	91.22%	93.09%	
	3	97.07%	2.93%	72.77%	94.23%	82.12%	
	Avg.	<b>97.87%</b>	<b>2.13%</b>	<b>88.97%</b>	<b>90.93%</b>	<b>89.94%</b>	
<b>Proposed</b>	1	99.43%	0.06%	99.15%	95.63%	97.36%	36.09 ms
	2	99.40%	0.06%	99.39%	96.07%	97.70%	
	3	99.75%	0.03%	98.13%	98.28%	98.20%	
	Avg.	<b>99.53%</b>	<b>0.05%</b>	<b>99.01%</b>	<b>96.42%</b>	<b>97.70%</b>	

Source: Created by the author (2015).



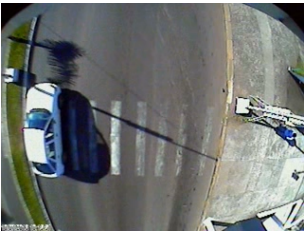


Table 4.7: Vehicle detection results for video sequence 3.

Video 3 results							
Method	Lane	Accuracy	Detection Error	Predictive Value	Sensitivity	F-Score	Time per frame
<b>Zivkovic (2006)</b>	1	98.57%	1.43%	87.20%	90.08%	88.62%	31.32 ms
	2	95.50%	4.50%	69.34%	86.07%	76.80%	
	3	94.79%	5.21%	60.09%	97.37%	74.32%	
	Avg.	<b>96.28%</b>	<b>3.72%</b>	<b>69.26%</b>	<b>91.04%</b>	<b>78.67%</b>	
<b>Noh (2013)</b>	1	98.52%	1.48%	85.74%	91.09%	88.33%	101 ms
	2	95.29%	4.71%	67.06%	89.60%	76.71%	
	3	94.17%	5.83%	57.35%	96.42%	71.92%	
	Avg.	<b>95.99%</b>	<b>4.01%</b>	<b>66.93%</b>	<b>92.35%</b>	<b>77.61%</b>	
<b>St-Charles (2014)</b>	1	98.74%	1.26%	88.13%	92.01%	90.03%	256.4 ms
	2	95.48%	4.52%	68.81%	87.44%	77.02%	
	3	94.73%	5.27%	59.80%	97.51%	74.14%	
	Avg.	<b>96.32%</b>	<b>3.68%</b>	<b>69.17%</b>	<b>92.15%</b>	<b>79.02%</b>	
<b>Method</b>	1	99.10%	0.90%	97.73%	87.32%	92.24%	45.09 ms
	2	97.99%	2.01%	92.22%	83.78%	87.79%	
	3	98.36%	1.64%	87.88%	91.29%	89.55%	
	Avg.	<b>98.48%</b>	<b>1.52%</b>	<b>92.01%</b>	<b>87.32%</b>	<b>89.60%</b>	

Source: Created by the author (2015).

Table 4.8: Vehicle detection and counting parameters used for each video sequence.

	Video 1	Video 2	Video 3
<b>Dataset</b>			
$\tau_{\gamma}^{\text{low}}$	0.94	0.9	0.91
$\tau_{\gamma}^{\text{high}}$	0.97	0.94	0.95
$\tau_l$	1	1	2
$\tau_u$	0.88	0.6	0.5
$\tau_{\mathcal{H}}$	0.25	0.5	0.35
$\tau_v$	2	2	4
$\tau_z$	0.6	0.6	0.6

Source: Created by the author (2015).

Table 4.9: Vehicle counting results for video sequence 1.

<b>Video 1 vehicle counting results</b>						
<b>Method</b>	<b>Lane</b>	<b>Count</b>	<b>Abs. Error</b>	<b>Error</b>	<b>FP</b>	<b>FN</b>
<b>Yuan, Zhao and Wang (2013)</b>	1	38	0	0%	3	3
	2	36	1	2.86%	1	0
	3	9	1	10.00%	1	2
	Total	<b>83</b>	<b>0</b>	<b>0%</b>	<b>5</b>	<b>5</b>
<b>Proposed Method</b>	1	38	0	0%	1	1
	2	35	1	2.86%	0	0
	3	10	1	10.00%	0	0
	Total	<b>83</b>	<b>0</b>	<b>0%</b>	<b>1</b>	<b>1</b>
<b>Ground-Truth</b>	1	38	-	-	-	-
	2	36	-	-	-	-
	3	9	-	-	-	-
	Total	<b>83</b>	-	-	-	-

Source: Created by the author (2015).

Table 4.10: Vehicle counting results for video sequence 2.

<b>Video 2 vehicle counting results</b>						
<b>Method</b>	<b>Lane</b>	<b>Count</b>	<b>Abs. Error</b>	<b>Error</b>	<b>FP</b>	<b>FN</b>
<b>Yuan, Zhao and Wang (2013)</b>	1	46	6	11.54%	3	9
	2	58	11	23.40%	10	0
	3	27	2	6.90%	3	4
	Total	<b>131</b>	<b>3</b>	<b>2.34%</b>	<b>16</b>	<b>13</b>
<b>Proposed Method</b>	1	46	6	11.54%	2	8
	2	55	8	17.02%	3	0
	3	25	4	13.79%	1	0
	Total	<b>126</b>	<b>2</b>	<b>1.56%</b>	<b>6</b>	<b>8</b>
<b>Ground-Truth</b>	1	52	-	-	-	-
	2	47	-	-	-	-
	3	29	-	-	-	-
	Total	<b>128</b>	-	-	-	-

Source: Created by the author (2015).

Table 4.11: Vehicle counting results for video sequence 3.

<b>Video 3 vehicle counting results</b>						
<b>Method</b>	<b>Lane</b>	<b>Count</b>	<b>Abs. Error</b>	<b>Error</b>	<b>FP</b>	<b>FN</b>
<b>Yuan, Zhao and Wang (2013)</b>	1	48	27	36.00%	10	37
	2	80	22	37.93%	34	12
	3	48	2	4.35%	13	11
	Total	<b>176</b>	<b>3</b>	<b>1.68%</b>	<b>57</b>	<b>60</b>
<b>Proposed Method</b>	1	65	10	13.33%	6	16
	2	61	3	5.17%	15	12
	3	55	9	19.57%	12	3
	Total	<b>181</b>	<b>2</b>	<b>1.12%</b>	<b>33</b>	<b>31</b>
<b>Ground-Truth</b>	1	75	-	-	-	-
	2	58	-	-	-	-
	3	46	-	-	-	-
	Total	<b>179</b>	-	-	-	-

Source: Created by the author (2015).

## 5 CONCLUSIONS AND DISCUSSION

Nowadays, the number of vehicles circulating in great urban centers is increasing. Therefore, traffic data have important applications in the development of cities. Traffic data can be used to plan urban development, reroute traffic to avoid traffic jams, synchronize traffic lights, etc. Many traffic monitoring systems use traditional sensors, such as microwave or magnetic inductive loops. With the advances in computer vision there is an ongoing research effort to replace traditional traffic monitoring sensors by image-based systems, which are easier to repair and are more flexible to operate. Image-based systems usually rely on computer vision techniques to detect the vehicles passing at specific locations, require less traffic interruptions for their maintenance (in comparison with traditional sensors), and can be operated remotely from traffic monitoring centers.

In this work, we proposed an image-based vehicular detection scheme for detecting vehicles at user-defined virtual loops, simulating the operation of traditional inductive loops. Using the proposed scheme, different user-defined virtual loops can be monitored simultaneously using one camera. Unfortunately, there are many challenging situations such as sudden illumination changes and cast shadows that can severely affect the results, resulting in inaccurate vehicular detections. Our proposed vehicle detection system is able to handle these situations. Initially, an approach based on Gaussian Mixture Models is used to obtain a foreground and a background model. The foreground and background models are used for detecting the vehicles passing at the user-defined virtual loops. To deal with the occurrence of shadows at the virtual loops, we combine the background model along with color and texture information to detect shadows at the virtual loops and to remove them before trying to detect vehicles. The background edges are detected using an efficient Gabor filters implementation, and the vehicles are detected at the virtual loops by comparing the detected background edges and the current frame edges. If the difference in terms of edge information at a virtual loop is significant, a vehicle is detected at that virtual loop. At each virtual loop, color and temporal vehicular detection information also are evaluated to further increase the robustness of the vehicle detection process. The proposed vehicular detection scheme outputs a vehicular detection signal for each monitored virtual loop, and this signal can be used for detecting and counting the vehicles passing at each virtual loop in a specific period of time.

We also proposed a novel hypergraph image segmentation shadow detection scheme, that provides good shadow detection and discrimination results. This method combines hypergraph image segmentation with color and texture information with a stochastic majority voting scheme

to obtain accurate shadow masks. First, the input image is segmented in multiple partitions using the hypergraph segmentation approach. This has the advantage of segmenting the image in multiple partitions and each partition contains pixels that are spatially close and have similar colors. Partitions that have average colors similar to the background are discarded (i.e. non-shadow partitions). Then, the input image is compared against a background model using color and texture information to obtain a rough shadow or non-shadow label for each pixel. Since this rough shadow label may contain misclassified pixels, this information is combined with the resulting partitionment obtained by the hypergraph image segmentation to validate or relabel these pixels. The validation step is performed by verifying if a partition consists mostly of pixels that were previously labeled as shadow pixels. In this case, every pixel in a partition is considered a shadow pixels. Since the hypergraph partitionment algorithms are stochastic, the image segmentation partitionment results may vary slightly. Therefore, to improve our detection results, for a given input image we perform multiple partitionments and shadow pixels labels. The final shadow mask is obtained by considering the pixels that were labeled as shadow pixels multiple times. Our proposed shadow detection scheme was not suitable for real-time processing, therefore, for the vehicle detection system, we decided to use a simpler and more efficient approach to keep the whole system as efficient as possible. Nonetheless, our proposed shadow detection scheme was capable of obtaining state-of-the art results when compared against recent and relevant shadow detection methods.

We performed experiments of both the proposed shadow detection and the vehicle detection schemes. For the proposed hypergraph image segmentation detection scheme, we performed experiments on seven common and publicly available shadow detection datasets and measured the detection rate, the discrimination rate and the average of both measures. Due to the stochasticity of the method, we measured the worst and best results and also the results with our proposed voting scheme. The experiments show that our proposed method was able to obtain more accurate results, in average, than all the comparative methods (87% detection rate and 89% discrimination rate). The video sequences used in the experiments consists of both indoor and outdoor scenes and also light and strong cast shadows. These results show that the proposed method is able to accurately detect shadows in different conditions.

For our vehicle detection system, our experiments were conducted with traffic videos containing challenging traffic monitoring situations (e.g. shadows, partial occlusions, illumination sudden changes, and long vehicles). The virtual loops were defined manually, and one virtual loop was placed at each road lane. Based on these experiments, the proposed scheme potentially can obtain a vehicular detection accuracy higher than 98% (in average). The processing

time per frame is comparable to the processing times of other methods representative of the state of the art. Higher processing times are expected as the presence of shadows increases at the virtual loops (since more processing is required for shadow detection and removal). Also, the experimental results suggest that our proposed method can obtain accurate vehicle counts, even in challenging situations (e.g. when shadows or sudden illumination changes tend to affect the results, and generate several false positives and false negatives). Therefore, based on these experiments, we may conclude that the proposed vehicular detection scheme potentially can be used for real-time vehicle detection and counting at user-defined virtual loops.

Future work will focus on improving the performance of our shadow detection scheme and integrating it with our vehicle detection system, as well as on conducting larger scale experiments with the proposed vehicular detection (and counting) scheme, involving several user-defined loops.

## 5.1 Contributions of this Work

In this section, we list the papers that resulted from our work, which were submitted to Journals:

### 5.1.1 Stochastic Shadow Detection Using a Hypergraph Partitioning Approach

#### Abstract

Shadows pose a challenging problem in computer vision applications. Shadow detection and removal methods are usually employed after foreground detection since shadows interfere with objects' segmentation, negatively affecting the results of tracking and motion detection algorithms. These methods have a trade-off between the detection rate and discrimination rate. We propose a new shadow detection method based on hypergraph segmentation that achieves good results in both the detection rate and discrimination rate. First, a weighted hypergraph of the image is constructed and partitioned. Then, we generate preliminary shadow and non-shadow masks using chromaticity and gradient correlation information. Finally, the preliminary shadow and non-shadow masks are used to classify the hypergraph partitions as shadows or non-shadows partitions. The resulting shadow mask is generated by combining all shadow partitions. With our approach, we were able to improve the state of the art in cast shadow detection methods.

#### Keywords

shadow detection, chromaticity, gradient correlation, hypergraph segmentation

#### Submitted to

Pattern Recognition

Type: Journal

Qualis: A1

Impact Factor (10/13/2015): 3.096

#### Status

Under review.



### **5.1.2 Novel Image-based Approach for Detecting Vehicles in User-defined Virtual Inductive Loops**

#### **Abstract**

Traffic monitoring systems are essential for managing modern urban centers and roadways. Nowadays, there is an ongoing research effort to replace traditional traffic monitoring sensors (e.g. inductive loops, microwave, etc.) by image-based systems, which are easier to repair and are more flexible to operate. We propose a new scheme that uses an image-based sensor (camera) for detecting vehicles at user-defined virtual loops, simulating the operation of inductive loops. Using the proposed scheme, several user-defined virtual loops can be monitored simultaneously with one camera. Initially, the background is modeled with an improved Gaussian Mixture Models approach, and then the shadows detected at the virtual loops are removed to minimize false vehicular detections. Next, vehicles are detected at the user-defined virtual loops using a combination of efficient edge detection and color information. The proposed scheme provides a vehicle detection signal that also can be used for counting vehicles at the user defined virtual loops. The experimental results indicate that the proposed scheme potentially can detect vehicles at user-defined virtual loops and provide their counts accurately (with more than 98% accuracy, in average), besides being more robust to cast shadows and sudden illumination changes than comparable methods that represent of the state of the art.

#### **Keywords**

vehicle detection, traffic monitoring, shadow detection, virtual loops, image-based sensors.

#### **Submitted to**

Information Sciences

Type: Journal

Qualis: A1

Impact Factor (10/13/2015): 4.038

#### **Status**

Under review.

# Appendices

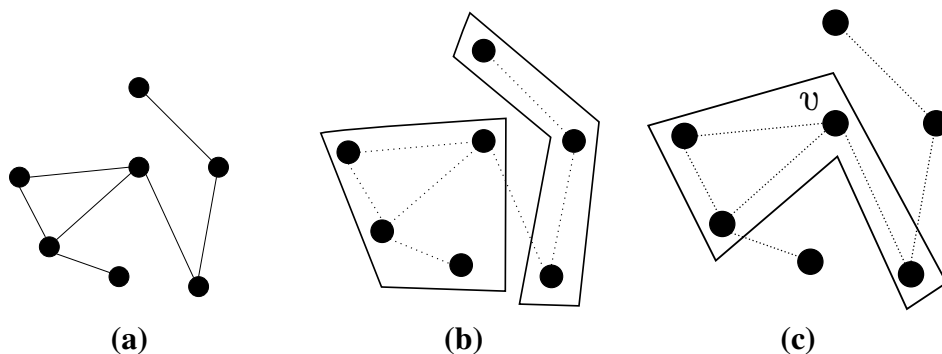
## Appendix A - FUNDAMENTALS OF IMAGE SEGMENTATION USING HYPERGRAPHS

We discuss here the theory behind hypergraph image segmentation, which are the basis of our hypergraph-based shadow detection scheme.

Hypergraphs are an extension of graphs in which edges are allowed to connect arbitrary, non-empty, sets of vertices (Rital; Miguet; Cherifi, 2005). Therefore, hypergraphs can model more general types of relations than graphs, and are used in image analysis applications involving data inter-dependencies (Rital; Miguet; Cherifi, 2005) (BRETTO; GILLIBERT, 2005) (BRETTO; CHERIFI, 2000) (KANNAN; KANNA; ARAVINDAN, 2010). Hypergraphs may be partitioned in a way to minimize a graph-cut cost function. The normalized cut method proposed by Shi and Malik (SHI; MALIK, 1997) is an alternative criterion which attempts to correct the tendency of the graph cut algorithms to favor isolated graph vertices. Rital, Miguet and Cherifi (2005) proposed a weighted adaptive approach for image segmentation using hypergraphs, which has the appeal of a strong theoretical basis. Most graph partitioning approaches try to find two sub-graphs  $A$  and  $B$  of a graph  $G$  that minimize a cost function  $cut\{A, B\} = \sum_{i \in A, j \in B} w(u, v)$ , where  $w(u, v)$  is the weight of the edge connecting the vertices  $u$  and  $v$ , with the following constraints:  $A \cup B = G$ ,  $A \cap B = \emptyset$  and  $A \neq \emptyset, B \neq \emptyset$ .

We used a modified version of the weighted adaptive image neighborhood approach proposed in (Rital; Miguet; Cherifi, 2005). More formally, a hypergraph is denoted by a pair  $H = (V, E)$ , where  $V = v_1, v_2, \dots, v_n$  is the set of vertices (or nodes) and  $E = E_1, E_2, \dots, E_m$ , where  $E_i \subseteq V$  and  $i = 1, \dots, m$ , is the set of hyperedges.

Figure A.1: Graph, hypergraph and neighborhood hypergraph example: (a) graph illustration; (b) hypergraph, and two sets of hypergraph vertices called hyperedges; (c) neighborhood hyperedge of the vertex  $v$  considering a distance of 1 (see Eq. (A.1) and Eq. (A.2)).



Source: Created by the author (2015).

Consider a graph  $G(V; e)$ , where  $e$  is the set of edges, and  $H(V, E)$  is the hypergraph with

the set of vertices of  $G$  and the hyperedges  $E$  representing the neighborhoods of these vertices, namely the *neighborhood hypergraph* of the graph  $G$  (see Fig. A.1). In fact, we can associate a neighborhood hypergraph to each graph  $G$  as follows:

$$H_G = (V, (E_v = \{v\} \cup \Gamma(v))), \quad (\text{A.1})$$

where

$$\Gamma(v) = \{u \in V, (v, u) \in e\}. \quad (\text{A.2})$$

A color image can be represented by the following mapping:

$$I : V \subseteq \mathbb{Z}^2 \rightarrow C \subseteq \mathbb{Z}^n. \quad (\text{A.3})$$

where the vertices of  $V$  are pixels and the elements of  $C$  are colors. A distance  $d$  on  $V$  defines a grid (a connected, regular graph, without loops or multi-edges). A distance  $d'$  can be defined on  $C$  in terms of the color similarity between two pixels. We then define the neighborhood relation in the image  $I$  by:

$$\Gamma_{\lambda, \beta}(v) = \{u \in V, |d'(I(v), I(u))| \leq \lambda \text{ and } d(v, u) \leq \beta\}, \quad (\text{A.4})$$

where the neighborhood of  $v$  on the grid is  $\Gamma_{\lambda, \beta}(v)$ . Consequently, we can associate an *Image Neighborhood Hypergraph (INH)* to an image  $I$  (Rital; Miguet; Cherifi, 2005) as follows:

$$H_{\Gamma_{\lambda, \beta}}(I) = (V, (v \cup \Gamma_{\lambda, \beta}(v))_{v \in V}). \quad (\text{A.5})$$

The predicate  $\lambda$  can be calculated adaptively (Rital; Miguet; Cherifi, 2005), but our experiments suggest that a fixed  $\lambda$  value also can be used. The hypergraph segmentation approach in (Rital; Miguet; Cherifi, 2005) was designed for grayscale images, so we modified the method in (Rital; Miguet; Cherifi, 2005) to work with color images. We first convert the image to the LAB color space and then use the  $\Delta E_{ab}^*$  distance (TKALCIC; TASIC, 2003), to estimate the similarity between two colors according to the human perception:

$$\Delta E_{ab}^* = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}, \quad (\text{A.6})$$

where  $L^*$ ,  $a^*$  and  $b^*$  are the color components in the LAB color space.

Given the neighborhood hypergraph of the input color image representing the foreground objects, we convert the image to grayscale and associate two weights to each vertex  $v_i$ :  $w_{v_i}$

and  $w_{h_i}$ . The weight  $w_{v_i}$  is the grayscale intensity of the pixel (i.e. vertex)  $v_i \in V$ , and the weight  $w_{h_i}$  is the mean grayscale intensity of the hyperedge  $h_i$  containing  $v_i$ . After constructing a neighborhood hypergraph for the foreground objects (see Fig. 3.1), we obtain  $K$  hypergraph partitions using the multilevel partitioning algorithm (KARYPIS et al., 1999).

The hypergraph partitioning problem is known to be NP-hard, and the algorithms most often used are heuristic (AYKANAT; CAMBAZOGLU; UÇAR, 2008). The multilevel paradigm for hypergraph partitioning (LASALLE; KARYPIS, 2013) has become popular since it has advantages as compared to other paradigms, like recursive bisection (RB), which recursively splits the hypergraph in two partitions using heuristics until the desired number of partitions is obtained. However, these heuristics may perform poorly when dealing with large hyperedges or small vertex degrees (the degree of a vertex is the number of hyperedges containing that vertex). Also, these heuristics tend to not provide a global view of the problem, often resulting in sub-optimal solutions (AYKANAT; CAMBAZOGLU; UÇAR, 2008). The multilevel paradigm addresses these issues and tends to produce high-quality hypergraph partitionings (DEVINE et al., 2006) (AYKANAT; CAMBAZOGLU; UÇAR, 2008) (LASALLE; KARYPIS, 2013).

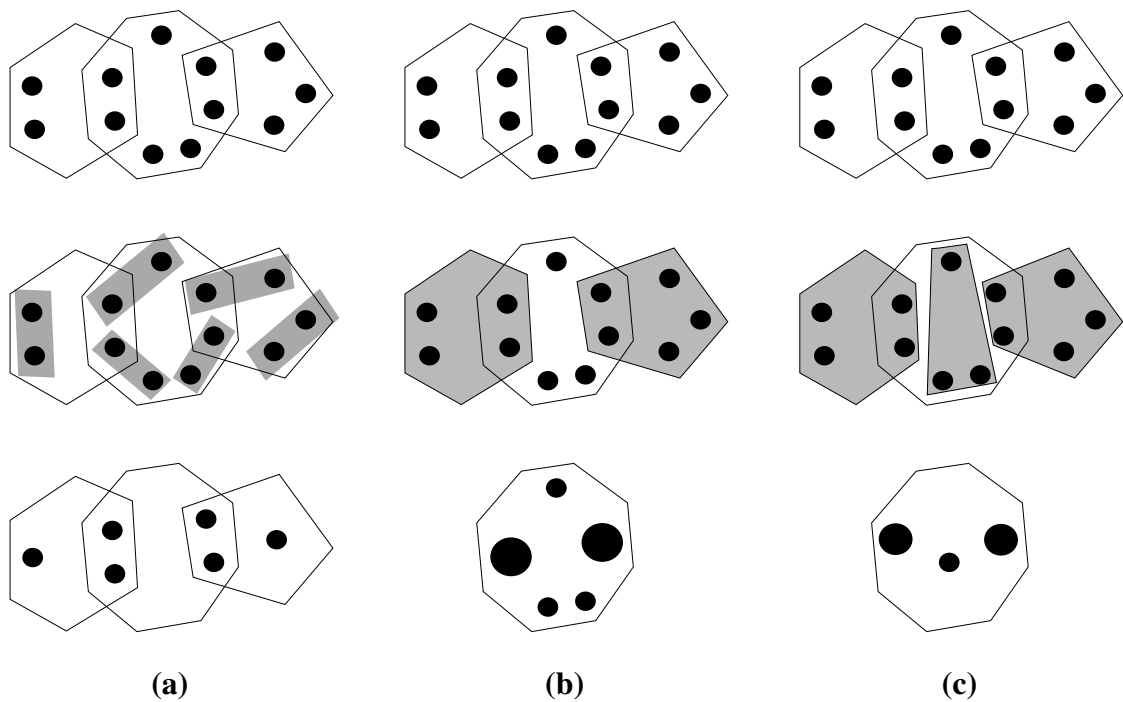
The multilevel approach for image hypergraph segmentation is divided in 3 main steps (LASALLE; KARYPIS, 2013): I) coarsening; II) initial hypergraph partitioning; and III) uncoarsening and refinement. In the coarsening step, a sequence of smaller (coarser) hypergraphs is constructed. These smaller (coarser) hypergraphs preserve the structure of the original hypergraph (i.e. the finer hypergraph) as accurately as possible (Rital; Miguet; Cherifi, 2005), and coarsening is achieved by merging vertices and contracting the hyperedges of the original hypergraph to form new hypergraph vertices in the coarser hypergraph (decreasing the size<sup>1</sup> of the hyperedges). Fig. A.2 illustrates three different approaches for the coarsening process. We tested these different coarsening approaches, and the hyperedge coarsening approach (see Fig. A.2(b)) provided the best experimental results (see Section 4.1).

In the next step of the multilevel approach for image hypergraph segmentation, the coarsest hypergraph is partitioned minimizing the cost of the hypergraph cut, which is followed by the uncoarsening step. The last step of the hypergraph image segmentation process is uncoarsening, and the coarsest hypergraph partition is expanded again to re-introduce the vertices of the original hypergraph in the obtained coarser hypergraph partitioning. Then, a hypergraph partition refinement algorithm is used to further improve this hypergraph partitioning, as discussed next. The steps of the multilevel approach for image hypergraph segmentation are summarized in Fig. A.3. The refinement algorithm reduces the cost of the hypergraph cut and improves the

---

<sup>1</sup>A hyperedge connecting  $k$  nodes has size  $k$ .

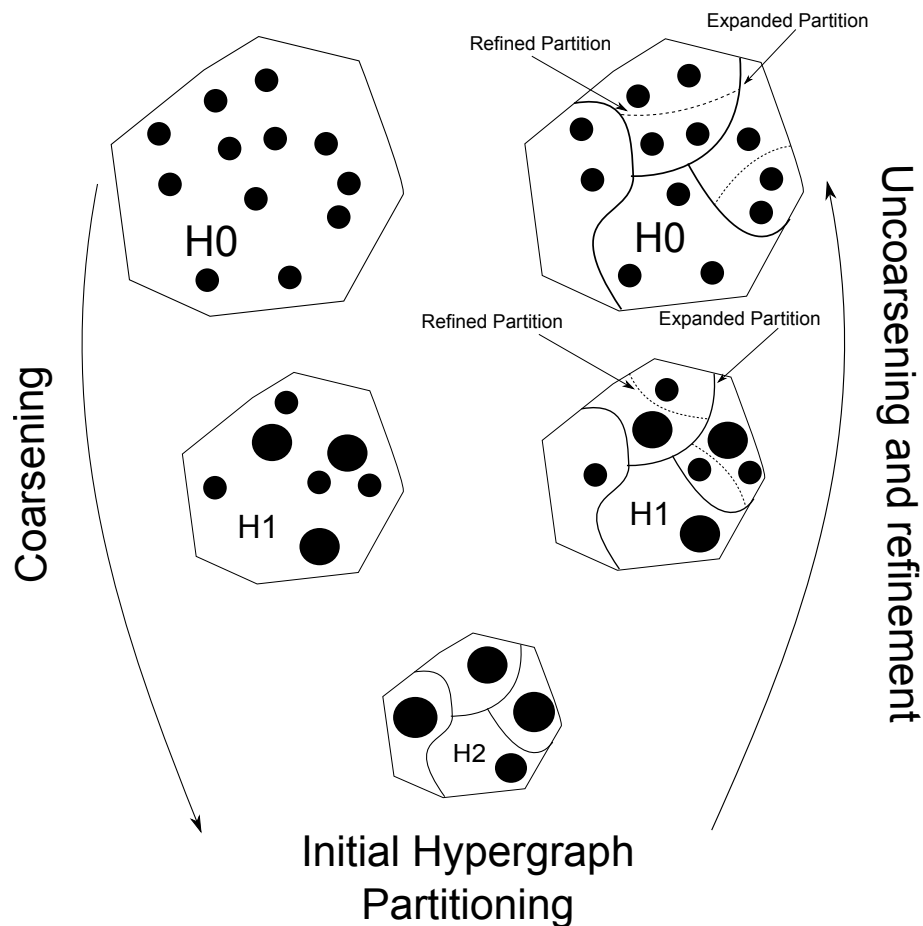
Figure A.2: Hypergraph coarsening illustration, top to bottom: **(a)** the hypergraph edge coarsening scheme in which pairs of connected vertices are merged together. The vertices are visited in random order and for each vertex  $v$ , all vertices that belong to hyperedges incident at  $v$  are considered, and the pairs of vertices connected via the largest weight edge are merged (KARYPIS; KUMAR, 1999); **(b)** hyperedge coarsening scheme in which a set of hyperedges is selected and the vertices that belong to these hyperedges are contracted together, giving preference to hyperedges with larger weights and smaller sizes (KARYPIS et al., 1999); and **(c)** modified hyperedge coarsening scheme, where after selecting the hyperedges that shall be contracted using the hyperedge coarsening scheme described before, all the uncontracted hyperedges vertices are considered in the contraction step. For each uncontracted hyperedge, the vertices that do not belong to any other hyperedge (that suffered contraction) are contracted together (KARYPIS et al., 1999).



Source: Created by the author (2015).

image hypergraph partitioning quality. Three refinement algorithms were evaluated for this task, namely: *FM* algorithm (FIDUCCIA; MATTHEYSES, 1982), which repeatedly moves vertices between partitions to reduce the cut cost (KARYPIS et al., 1999); *Early-Exit FM*, which is a simplified version of the FM algorithm that defines a maximum number of passes to be performed, and each pass stops if after moving  $k$  vertices the cut cost is reduced (KARYPIS et al., 1999); and *Hyperedge Refinement (HER)*, that iteratively moves groups of vertices between partitions until an entire hyperedge is removed from the hypergraph cut set (KARYPIS et al., 1999). In our experiments, the FM algorithm provided the best results and is used in this work as the image hypergraph partitioning refinement method.

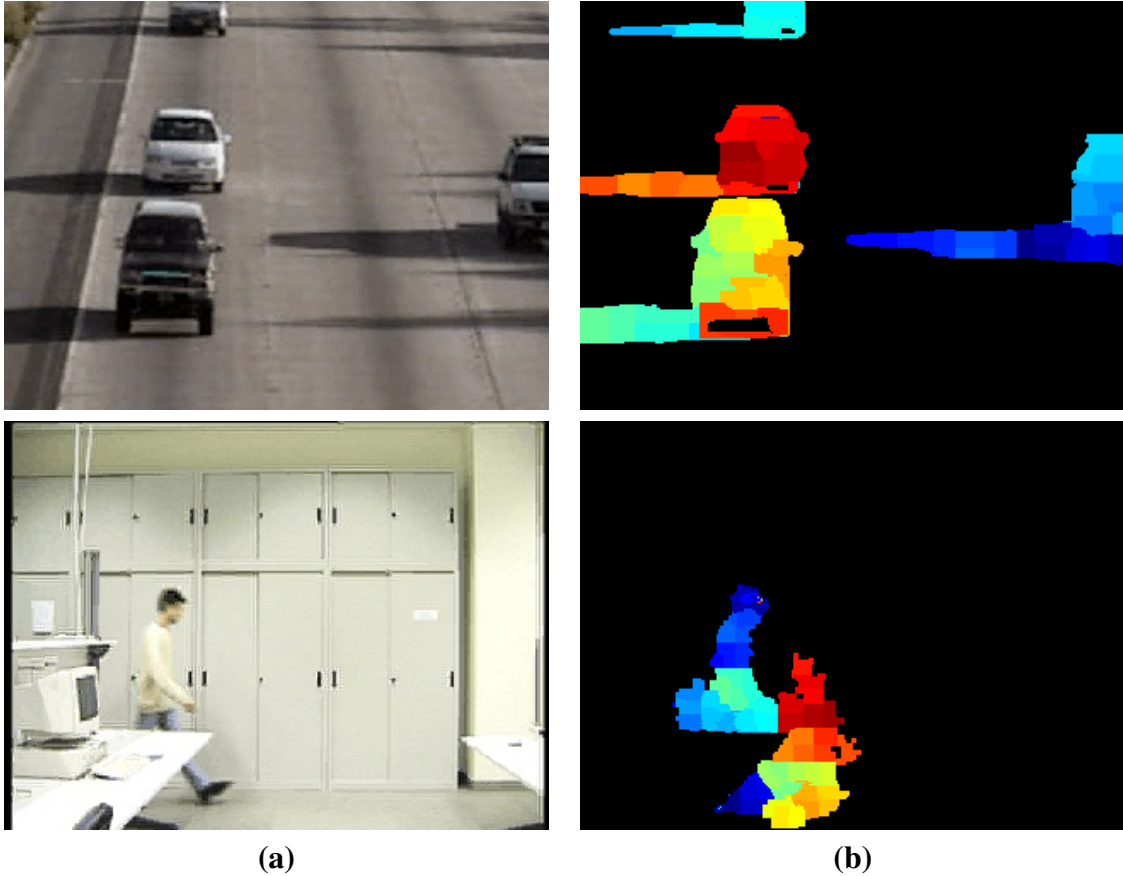
Figure A.3: Illustration of the 3 steps of the multilevel approach used in this work for image hypergraph segmentation (Rital; Miguet; Cherifi, 2005).



Source: Created by the author (2015).

An illustration of the proposed image segmentation scheme based on image hypergraph partitioning can be seen in Fig. A.4. Since we are only interested in the foreground objects shadows, the foreground mask provided as input to our method is used to build the image

Figure A.4: Hypergraph segmentation results for a frame of the Highway dataset, and for a frame of the Lab dataset: **(a)** original frame; **(b)** image hypergraph partitioning result. Each color represents an image hypergraph partition corresponding to a foreground region, and black regions correspond to the background.



Source: Created by the author (2015).

hypergraph, which will be used to detect shadows regions in the foreground (see Fig. 3.1).



## REFERENCES

- AL-NAJDAWI, N. et al. A survey of cast shadow detection algorithms. **Pattern Recognition Letters**, Elsevier B.V., v. 33, n. 6, p. 752–764, abr. 2012. ISSN 01678655.
- AYKANAT, C.; CAMBAZOGLU, B. B.; UÇAR, B. Multi-level direct k-way hypergraph partitioning with multiple constraints and fixed vertices. **Journal of Parallel and Distributed Computing**, v. 68, n. 5, p. 609 – 625, 2008. ISSN 0743-7315.
- BARCELLOS, P. et al. A novel video based system for detecting and counting vehicles at user-defined virtual loops. **Expert Systems with Applications**, Elsevier Ltd, n. October, oct. 2014. ISSN 09574174.
- BARNICH, O.; DROOGENBROECK, M. V. Vibe: A universal background subtraction algorithm for video sequences. **Image Processing, IEEE Transactions on**, v. 20, n. 6, p. 1709–1724, June 2011. ISSN 1057-7149.
- BENEZETH, Y. et al. Comparative study of background subtraction algorithms. **Journal of Electronic Imaging**, v. 19, n. 3, p. 03–33, 2010.
- BRETTO, A.; CHERIFI, H. Noise detection and cleaning by hypergraph model. In: **Proceedings of the International Conference on Information Technology: Coding and Computing**. [S.l.: s.n.], 2000. p. 416–419.
- BRETTO, A.; GILLIBERT, L. Hypergraph-based image representation. In: BRUN, L.; VENTO, M. (Ed.). **Graph-Based Representations in Pattern Recognition, Proceedings of the 5th IAPR International Workshop (GbRPR 2005), Poitiers, France, April 11-13, 2005**. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3434), p. 1–11. ISBN 3-540-25270-3.
- BRIECHLE, K.; HANEBECK, U. D. Template matching using fast normalized cross correlation. In: **Proceedings of SPIE: Optical Pattern Recognition XII**. [S.l.: s.n.], 2001. v. 4387, p. 95–102.
- BULLKICH, E. et al. Moving shadow detection by nonlinear tone-mapping. In: **19th International Conference on Systems, Signals and Image Processing (IWSSIP)**. [S.l.: s.n.], 2012. p. 146–149. ISSN 2157-8672.
- CHOI, W.-P. et al. Simplified Gabor wavelets for human face recognition. **Pattern Recognition**, v. 41, p. 1186–1199, 2008. ISSN 00313203.
- COMANICIU, D.; MEER, P. Mean shift: a robust approach toward feature space analysis. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 24, n. 5, p. 603–619, May 2002. ISSN 0162-8828.
- CUCCHIARA, R. et al. Detecting moving objects, ghosts, and shadows in video streams. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 25, n. 10, p. 1337–1342, oct. 2003. ISSN 0162-8828.
- DEVINE, K. et al. Parallel hypergraph partitioning for scientific computing. In: **20th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2006**. [S.l.: s.n.], 2006. p. 1–10.

- FIDUCCIA, C.; MATTHEYSES, R. A linear-time heuristic for improving network partitions. **19th Conference on Design Automation, 1982.**, p. 175–181, June 1982. ISSN 0146-7123.
- GANGODKAR, D.; KUMAR, P.; MITTAL, A. Robust Segmentation of Moving Vehicles Under Complex Outdoor Conditions. **IEEE Transactions on Intelligent Transportation Systems**, v. 13, n. 4, p. 1738–1752, dec. 2012. ISSN 1524-9050, 1558-0016.
- GUO, R.; DAI, Q.; HOIEM, D. Single-image shadow detection and removal using paired regions. In: **Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)**. Washington, DC, USA: IEEE Computer Society, 2011. p. 2033–2040. ISBN 978-1-4577-0394-2.
- HEO, Y. S.; LEE, K. M.; LEE, S. U. Robust stereo matching using adaptive normalized cross-correlation. **IEEE transactions on pattern analysis and machine intelligence**, v. 33, n. 4, p. 807–22, abr. 2011. ISSN 1939-3539.
- HIRSCHMULLER, H.; SCHARSTEIN, D. Evaluation of cost functions for stereo matching. In: **IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07**. [S.l.: s.n.], 2007. p. 1–8. ISSN 1063-6919.
- HMETIS - Hypergraph and Circuit Partitioning library.  
[Http://glaros.dtc.umn.edu/gkhome/metis/hmetis/overview](http://glaros.dtc.umn.edu/gkhome/metis/hmetis/overview), visited on: 15 nov. 2015.
- HSIEH, J.-W. et al. Shadow elimination for effective moving object detection by gaussian shadow modeling. **Image and Vision Computing**, v. 21, n. 6, p. 505 – 516, 2003. ISSN 0262-8856.
- HUANG, J.-B.; CHEN, C.-S. Moving cast shadow detection using physics-based features. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)**. [S.l.: s.n.], 2009. p. 2310–2317. ISSN 1063-6919.
- JIANG, W.; LAM, K.-M.; SHEN, T.-Z. Efficient edge detection using simplified gabor wavelets. **IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society**, v. 39, n. 4, p. 1036–47, aug. 2009. ISSN 1941-0492.
- JUNG, C.; SCHARCANSKI, J. Robust watershed segmentation using the wavelet transform. In: **Proceedings of the XV Brazilian Symposium on Computer Graphics and Image Processing**. [S.l.: s.n.], 2002. p. 131–137. ISSN 1530-1834.
- JUNG, C.; SCHARCANSKI, J. Wavelet transform approach to adaptive image denoising and enhancement. **Journal of Electronic Imaging**, v. 13, n. 2, p. 278–285, 2004.
- JUNG, C.; SCHARCANSKI, J. Robust watershed segmentation using wavelets. **Image and Vision Computing**, v. 23, n. 7, p. 661 – 669, 2005. ISSN 0262-8856.
- KANNAN, K.; KANNA, B. R.; ARAVINDAN, C. Root mean square filter for noisy images based on hyper graph model. **Image and Vision Computing**, v. 28, n. 9, p. 1329 – 1338, 2010. ISSN 0262-8856.
- KARYPIS, G. et al. Multilevel hypergraph partitioning: applications in vlsi domain. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 7, n. 1, p. 69–79, March 1999. ISSN 1063-8210.

- KARYPIS, G.; KUMAR, V. Multilevel k-way hypergraph partitioning. In: **Proceedings of the 36th Annual ACM/IEEE Design Automation Conference**. New York, NY, USA: ACM, 1999. (DAC '99), p. 343–348. ISBN 1-58113-109-7.
- KHARE, M.; SRIVASTAVA, R. K.; KHARE, A. Dual tree complex wavelet transform based shadow detection and removal from moving objects. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **IS&T/SPIE Electronic Imaging**. [S.l.], 2014. p. 90–97.
- LASALLE, D.; KARYPIS, G. Multi-threaded graph partitioning. In: **Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing**. Washington, DC, USA: IEEE Computer Society, 2013. (IPDPS '13), p. 225–236. ISBN 978-0-7695-4971-2.
- LEWIS, J. P. Fast normalized cross-correlation. 1995.  
[Http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.6062](http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.6062), visited on: 15 nov. 2015.
- NOH, S.; JEON, M. A New Framework for Background Subtraction Using Multiple Cues. **Computer Vision–ECCV 2012**, n. 1, p. 493–506, 2013.
- OTSU, N. A Threshold Selection Method from Gray-level Histograms. **IEEE Transactions on Systems, Man and Cybernetics**, v. 9, n. 1, p. 62–66, 1979.
- PELLEGRINO, F. A.; VANZELLA, W.; TORRE, V. Edge detection revisited. **IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics**, v. 34, n. 3, p. 1500–1518, 2004. ISSN 10834419.
- PORNANOMCHAI, C.; LIAMSANGUAN, T.; VANNAKOSIT, V. Vehicle detection and counting from a video frame. **2008 International Conference on Wavelet Analysis and Pattern Recognition**, IEEE, p. 356–361, aug. 2008.
- POURMOHAMMAD, A.; FALLAHPOUR, M.; KARIMIFAR, S. Scene matching based on using edge features. In: **2012 8th International Conference on Information Science and Digital Content Technology (ICIDT)**. [S.l.: s.n.], 2012. v. 1, p. 189–194.
- PRATI, A. et al. Analysis and detection of shadows in video streams: a comparative evaluation. In: **Proceedings of the 2001 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)**. [S.l.: s.n.], 2001. v. 2, p. II–571–II–576 vol.2. ISSN 1063-6919.
- Rital, S.; Miguet, S.; Cherifi, H. Weighted Adaptive Neighborhood Hypergraph Partitioning for Image Segmentation. In: SINGH, S. et al. (Ed.). **3rd International Conference on Advances in Pattern Recognition**. [S.l.]: LNCS, 2005. (Pattern Recognition and Image Analysis). ISBN 3-540-28833-3.
- SANCHEZ, A. et al. Video-based distance traffic analysis: Application to vehicle tracking and counting. **Computing in Science Engineering**, v. 13, n. 3, p. 38–45, May 2011. ISSN 1521-9615.
- SANIN, A.; SANDERSON, C.; LOVELL, B. C. Improved Shadow Removal for Robust Person Tracking in Surveillance Scenarios. **20th International Conference on Pattern Recognition**, IEEE, p. 141–144, aug. 2010.
- SANIN, A. et al. Shadow detection: A survey and comparative evaluation of recent methods. **Pattern Recognition**, v. 45, n. 4, p. 1684–1695, abr. 2012. ISSN 00313203.

SCHARCANSKI, J. et al. A Particle-Filtering Approach for Vehicular Tracking Adaptive to Occlusions. **IEEE Transactions on Vehicular Technology**, v. 60, n. 2, p. 381–389, feb. 2011. ISSN 0018-9545.

SEBASTIAN, P. Tracking using normalized cross correlation and color space. **2007 International Conference on Intelligent and Advanced Systems**, IEEE, p. 770–774, nov. 2007.

SHABANINIA, E.; NAGHSH-NILCHI, A. Robust watershed segmentation of moving shadows using wavelets. In: **8th Iranian Conference on Machine Vision and Image Processing (MVIP)**. [S.l.: s.n.], 2013. p. 381–386. ISSN 2166-6776.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, p. 888–905, 1997.

SILVA, L.; SCHARCANSKI, J. Video segmentation based on motion coherence of particles in a video sequence. **IEEE Transactions on Image Processing**, v. 19, n. 4, p. 1036–1049, April 2010. ISSN 1057-7149.

SIVARAMAN, S.; TRIVEDI, M. M. Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. **IEEE Transactions on Intelligent Transportation Systems**, p. 1–23, 2013. ISSN 1524-9050.

SOBRAL, A. BGSLibrary: An opencv c++ background subtraction library. In: **IX Workshop de Visão Computacional (WVC'2013)**. Rio de Janeiro, Brazil: [s.n.], 2013. <https://github.com/andrewssobral/bgslibrary>, visited on: 15 nov. 2015.

ST-CHARLES, P.-L.; BILODEAU, G.-a. Improving background subtraction using Local Binary Similarity Patterns. **IEEE Winter Conference on Applications of Computer Vision**, p. 509–515, 2014.

STAUFFER, C.; GRIMSON, W. Adaptive background mixture models for real-time tracking. In: **Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on**. [S.l.: s.n.], 1999. v. 2, p. 252 Vol. 2. ISSN 1063-6919.

TIAN, B. et al. Video processing techniques for traffic flow monitoring: A survey. **2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)**, IEEE, p. 1103–1108, oct. 2011.

TKALCIC, M.; TASIC, J. Colour spaces: perceptual, historical and applicational background. In: **The IEEE Region 8 EUROCON 2003, Computer as a Tool**. [S.l.: s.n.], 2003. v. 1, p. 304–308 vol.1.

XU, D. et al. Insignificant shadow detection for video segmentation. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 15, n. 8, p. 1058–1064, Aug 2005. ISSN 1051-8215.

YUAN, Y.; ZHAO, Y.; WANG, X. Day and Night Vehicle Detection and Counting in Complex Environment. **2013 28th International Conference on Image and Vision Computing New Zealand (IVCNZ 2013)**, IEEE, p. 453–458, nov. 2013.

ZIVKOVIC, Z. Improved adaptive gaussian mixture model for background subtraction. In: **Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.** [S.l.: s.n.], 2004. v. 2, p. 28–31 Vol.2. ISSN 1051-4651.

ZIVKOVIC, Z.; HEIJDEN, F. van der. Efficient adaptive density estimation per image pixel for the task of background subtraction. **Pattern Recognition Letters**, v. 27, n. 7, p. 773–780, may 2006. ISSN 01678655.

ZWENG, A.; RITTLER, T.; KAMPEL, M. Evaluation of histogram-based similarity functions for different color spaces. **Computer Analysis of Images and Patterns**, p. 455–462, 2011.