Efficient embedded context-based surveillance image and video analysis

Citation for published version (APA):
Javanbakhti, S. (2019). *Efficient embedded context-based surveillance image and video analysis*. [Phd Thesis 1 (Research TU/e / Graduation TU/e), Electrical Engineering]. Technische Universiteit Eindhoven.

Document status and date:
Published: 25/06/2019

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Download date: 04. Oct. 2023

Efficient embedded context-based surveillance image and video analysis

Efficient embedded context-based surveillance image and video analysis

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit
Eindhoven, op gezag van de rector magnificus prof.dr.ir. F.P.T. Frank
Baaijens, voor een commissie aangewezen door het College voor Promoties,
in het openbaar te verdedigen op 25 juni 2019 om 16:00 uur

door

Solmaz Javanbakhti

geboren te Teheran, Iran

Dit proefschrift is goedgekeurd door de promotor en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof.dr.ir. A.M.J. Koonen
1e promotor:	prof.dr.ir. Peter H.N. de With
copromotor(en):	dr. S. Zinger
leden:	prof.dr. Th. Gevers (Universiteit van Amsterdam)
	prof.dr.ir. R.N.J. Veldhuis (Universiteit Twente)
	prof.dr.ir. G. de Haan
Adviseur(s):	dr.ir. R.G.J. Wijnhoven (ViNotion BV)

Efficient embedded context-based surveillance image and video analysis
Solmaz Javanbakhti.
A catalogue record is available from the Eindhoven University of Technology Library
ISBN: 978-94-028-1573-3
NUR: 959

Cover design: Leily Khoshgoftar
Printed by: IpsKamp Drukkers

Copyright © 2019 by S. Javanbakhti

All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic, mechanical, including photocopying, recording or by any information storage and retrieval system, without the prior permission of the copyright owner.

To my love Pooya for his endless support, my dear parents for their love and encouragement, and to my little angle Lily-Rose.

Summary

Efficient embedded context-based surveillance image and video analysis

Visual surveillance in dynamic scenes has a wide range of potential applications, such as security for important buildings and traffic surveillance in cities and highways, detection of military targets, etc. Surveillance systems can perform automated event analysis using embedded video analysis modules. Automated surveillance may reduce labor costs and decreases the chance of missing an important event in one of the many videos in the system. Smart surveillance systems should be able to at least detect and track moving objects, classify these objects and interpret their activities. Although such systems have been widely explored, they still need improvements in terms of reliability and robustness with respect to the semantic understanding and interpretation of scenes. These improvements can be realized by including contextual information in the video analysis that involves the relationship between a scene and its enclosed objects and regions (e.g. in airport surveillance, man-made objects like abandoned suitcases provide such contextual information). Besides this, the extra context analysis should be embedded into other algorithms, so that algorithm complexity and efficiency are improved to limit system costs. This thesis exploits the use of contextual information to achieve a better interpretation of events based on object behavior with higher reliability and robustness, and also a higher semantic level of scene understanding by adding contextual information about the scene itself. Context is explored at different levels: (1) feature information, such as color, texture, edge, motion, (2) spatial region properties, such as salient region and semantic labeled region, (3) semantically meaningful information

on events, like specific object behavior, such as jointly moving objects or abnormal behavior. We have developed multiple generic and fast approaches (e.g. semantic region labeling, salient region detection, motion analysis) to extract contextual information. The algorithms are designed to be embedded as a processing block in another larger algorithm or another application (e.g. labelled water region as context to be used in an advanced ship detection algorithm). The first contributing part of this thesis in Chapter 3 aims at salient region detection and explores four detection techniques based on various features: (1) sum of edge pixels, (2) number of connected components, (3) number of straight lines and (4) the entropy of the frequency-domain features of a DCT. It is shown that the proposed DCT-based salient region detection technique is a valuable approach. Subsequently, Chapter 4 considers semantic region labeling in two distinct ways. The first option is labeling of each individual specific region, e.g., separate approaches to label sky, water and road regions, while the second way develops a general framework for labeling an input scene with multiple semantic labels such as sky, road and vegetation at the same time. The chapter concludes that developing specific algorithms for each specific region is not adaptive to new region types and would need to re-design a new approach for each new region type. Therefore, our research study for semantic region labeling has resulted into proposing a general framework for performing automatic semantic labeling of video scenes by combining the local features and spatial contextual cues. We have demonstrated that our region labeling is more accurate than a relevant proposal in literature. Chapter 5 studies complexity of the algorithms proposed in Chapter 3 and Chapter 4 in detail to evaluate possible implementation in (embedded, real-time) video systems. The performed complexity analysis method is based on counting native DSP operations and memory transfers assuming a basic RISC CPU as a reference model. It is concluded from Chapters 3, 4 and 5 that the proposed contextual information extraction methods quantitatively and qualitatively outperform state-of-the-art approaches in accuracy while having a lower complexity, making these methods valuable. The last thesis part covered in Chapter 6 presents four use cases where the proposed techniques are validated by applying them into multiple complex surveillance situations. First, the detection of moving ships in harbor surveillance uses the semantic region labeling approach from Chapter 4 and combines motion context information to detect moving ships reliably and in a robust way in port surveillance videos. Second, the recognition of traffic action employs our region labeling algorithm from Chapter 4 and combines automatic traffic-sign information, to recognize actions in traffic surveillance video. It is proven that the proposed traffic action recognition system evaluations works well for the dataset at hand due to the considerable testing of the individual components and the relative simplicity of the decision engines for the scenario content. Third, detection of moving cars in traffic surveillance involves two sub-cases to detect moving cars in traffic surveillance videos. In the first sub-case, the semantic region labeling algorithm from Chapter 4 is combined with motion information. We show that our system with more context information does not generate false positive vehicles and misses fewer occluded cars, compared to literature. In the second sub-case, our DCT-based salient region detection technique from Chapter 3 is again combined with motion information. It is indicated that, although the detection rate is somewhat reduced, the proposed framework leads to a better selective usage of a computationally expensive car detector, thereby making the approach more efficient. Fourth, fast abnormal event detection is explored with a novel block-based approach based on analyzing the pixel-based motion context, as an alternative for the conventional object-based approach. We have discovered that the entropy of the DCT-transformed motion magnitude is a reliable measure for classifying whether the current activity in the video is normal or not. Our framework is generic and does not depend on the type of scene. At the end of Chapter 6, we compare our region labeling with a recently developed region labeling system based on emerging deep learning technology. By applying the algorithms to two different datasets, we conclude that our algorithm performs -as expected- on the average slightly lower than deep learning, but with a lower computational complexity. Overall, the use cases provide evidence that embedded context information helps to obtain a better semantic event interpretation occurring in a monitored space.

In conclusion, this thesis shows that contextual information is not complicated to extract from a surveillance video, while it adds significant value to automated surveillance analysis, or it enables automated analysis of complicated scenarios that were previously not possible or too expensive with conventional object techniques. The presented concepts of using context information at different levels offer a higher level of understanding or a better robustness. Emerging technologies for better context extraction in the future can be based on thermal infrared sensing. With respect to algorithm development, it is evident that deep learning and convolutional neural networks (CNNs) are playing already an important role for making surveillance systems more intelligent and robust, but the embedding complexity aspects and efficiency of these new tools clearly require further study to limit system costs.

Samenvatting

Visuele bewaking in dynamische scènes heeft een breed scala aan mogelijke toepassingen, zoals beveiliging van belangrijke gebouwen en verkeerssurveillance in steden en snelwegen, detectie van militaire doelen, enz. Surveillancesystemen kunnen geautomatiseerde analyses uitvoeren op gebeurtenissen met diverse ingebedde videomodules. Geautomatiseerd toezicht kan de arbeidskosten verlagen en verkleint de kans op het missen van een belangrijke gebeurtenis in een van de vele video's in het systeem. Slimme bewakingssystemen moeten op zijn minst bewegende objecten kunnen detecteren en volgen, deze objecten kunnen classificeren en hun activiteiten kunnen interpreteren. Hoewel dergelijke systemen uitgebreid zijn onderzocht, hebben ze nog steeds verbeteringen nodig t.a.v. de betrouwbaarheid en robuustheid van de semantische interpretatie van scènes. Deze verbeteringen kunnen worden gerealiseerd door contextuele informatie op te nemen in de videoanalyse die betrekking heeft op de relaties tussen een scène en de daarin aanwezige objecten en gebieden (bijv. bij luchthavenbewaking geven de mensgerelateerde objecten zoals verlaten koffers dergelijke contextuele informatie). Daarnaast moet de extra contextanalyse worden geïntegreerd in andere algoritmen, zodat de complexiteit en efficiëntie van de algoritmen worden verbeterd en systeemkosten worden beperkt. Dit onderzoek gebruikt contextinformatie om een betere interpretatie van gebeurtenissen te bereiken op basis van objectgedrag met een hogere betrouwbaarheid en robuustheid, en ook om een hoger semantisch niveau van scènebegrip te krijgen door contextuele informatie over de scène zelf toe te voegen. Context wordt op verschillende niveau's verkend: (1) functie-informatie, zoals kleur, textuur, vorm, beweging, (2) eigenschappen van beeldgebieden (regio's), zoals contrastrijke textuur en semantisch gelabelde gebieden, (3) semantisch zinvolle informatie over het gedrag of gebeurtenis, zoals specifiek objectgedrag (bijv. gezamenlijk bewegende objecten of abnormaal gedrag). Er zijn verscheidene generieke en snelle algoritmen ontwikkeld (bijv. semantisch regiolabelen, detectie van opvallende regio's, bewegingsanalyse) om contextuele informatie te extraheren. De algoritmen zijn ontworpen om te worden ingebed als een verwerkingseenheid in een groter algoritme of een andere toepassing (zoals gelabeld watergebied als context voor een geavanceerde detectie van schepen). De eerste bijdrage van dit proefschrift in Hoofdstuk 3 richt zich op de detectie van opvallende gebieden en onderzoekt vier detectietechnieken op basis van verschillende kenmerken: (1) som van de pixels op een objectrand, (2) aantal verbonden componenten, (3) aantal rechte lijnen en (4) de entropie van de frequentiekarakterisatie met een DCT. Er wordt aangetoond dat de DCT-gebaseerde detectietechniek voor opvallende regio's een goede benadering is. Vervolgens wordt in Hoofdstuk 4 het semantisch regiolabelen op twee verschillende manieren beschouwd. De eerste optie is het labelen van elk afzonderlijk gebied, bijv. door afzonderlijke labels te geven voor lucht-, water- en wegregio's, terwijl de tweede manier een algemeen kader ontwikkelt voor het labelen van een scène met verschillende semantische labels zoals lucht, weg en vegetatie op hetzelfde tijdsmoment. De conclusie is dat het ontwikkelen van specifieke algoritmen voor elke aparte regio niet geschikt is voor nieuwe gebieden, omdat voor elk nieuw regiotype opnieuw een ontwerp nodig is. Daarom heeft het onderzoek voor het markeren van semantische regio's geleid tot een algemeen kader voor het uitvoeren van automatisch labelen van gebieden in videoscènes, waarbij de lokale kenmerken en spatiële contextuele aanwijzingen worden gecombineerd. Het is aangetoond dat deze methode voor labelen van regio's nauwkeuriger is dan een relevant voorstel uit de literatuur. Hoofdstuk 5 bestudeert in detail de complexiteit van de algoritmen die zijn ontworpen in de Hoofdstukken 3 en 4 om mogelijke implementatie in (ingebedde, real-time) videosystemen te evalueren. De uitgevoerde methode voor complexiteitsanalyse is gebaseerd op het tellen van intrinsieke DSP-bewerkingen en acties voor geheugenopslag, met een standaard RISC-CPU als referentiemodel. Uit de Hoofdstukken 3, 4 en 5 wordt geconcludeerd dat de ontworpen methodes voor contextuele informatie-extractie kwantitatief en kwalitatief beter presteren dan de gepubliceerde algoritmen qua nauwkeurigheid, terwijl ze een lagere complexiteit hebben. Het laatste deel van het proefschrift in Hoofdstuk 6 beschrijft vier gevallen waarin de ontwikkelde technieken gevalideerd worden door ze toe te passen in diverse complexe surveillance-situaties. In het eerste geval gebruikt de detectie van bewegende schepen in havenbewaking het semantische regi-

olabelen uit Hoofdstuk 4 en combineert bewegingsinformatie als context om bewegende schepen betrouwbaar en op een robuuste manier te detecteren in bewakingsvideo's. Het tweede geval over verkeersgedrag exploiteert het ontworpen regiolabelen uit Hoofdstuk 4 en combineert dit met automatisch gevonden verkeersbordinformatie om gedrag in een bewakingsvideo te herkennen. Experimentele evaluaties tonen aan dat de herkenning van verkeersgedrag goed werkt voor de betreffende dataset vanwege het intensief testen van de afzonderlijke componenten en de relatieve eenvoud van de beslissingseenheid voor gedragsanalyse. Het derde geval omvat het detecteren van bewegende auto's bij verkeerscontrole met daarin twee deelgevallen om bewegende auto's te detecteren in verkeersbeveiliging. In het eerste deelgeval wordt het algoritme voor het markeren van semantische gebieden uit Hoofdstuk 4 gecombineerd met bewegingsinformatie. Het systeem met meer contextinformatie genereert geen foute voertuigdetecties en mist minder auto's die nog niet waren gedetecteerd, in vergelijking met systemen uit de literatuur. In het tweede deelgeval wordt de DCT-gebaseerde methode voor regiodetectie uit Hoofdstuk 3 opnieuw gecombineerd met bewegingsinformatie. Hoewel de detectiesnelheid enigszins is verlaagd, leidt het nieuwe voorstel tot een beter selectief gebruik van een rekenintensieve detectie, waardoor het geheel efficiënter wordt. Als vierde geval wordt een snelle abnormale gebeurtenis gedetecteerd met een nieuwe, blokgebaseerde aanpak met daarin het analyseren van de pixelgebaseerde bewegingscontext, als een alternatief voor de conventionele objectgebaseerde benadering. Er wordt gevonden dat de entropie van de DCT-getransformeerde bewegingswaarde een betrouwbare maat is om te classificeren of de huidige activiteit in de video normaal is of niet. Het ontwikkelde systeem is generiek en hangt niet af van het type scène. Aan het einde van Hoofdstuk 6 wordt het ontworpen regiolabelen vergeleken met een vergelijkbaar nieuw ontwikkeld systeem op basis van Deep Learning technologie door de algoritmen toe te passen op twee verschillende datasets. De conclusie is dat het algoritme uit het onderzoek –zoals verwacht– gemiddeld iets lager scoort dan Deep Learning, maar met een lagere rekenkundige complexiteit. Over het algemeen leveren de onderzochte situaties het bewijs dat inbedden van contextinformatie helpt bij het verkrijgen van een betere semenatische interpretatie van de gebeurtenissen in een bewaakt gebied.
indent Concluderend laat dit proefschrift zien dat contextuele informatie niet ingewikkeld is om te extraheren uit een bewakingsvideo, terwijl het significante waarde toevoegt aan geautomatiseerde surveillance-analyse. Daarnaast maakt het geautomatiseerde analyse van ingewikkelde scenario's mogelijk die voorheen niet mogelijk waren of te duur zijn voor een conventioneel herkenningssysteem. De gepresenteerde concepten voor het gebruiken

van contextinformatie op verschillende niveau's bieden een hoger begripsniveau of een betere robuustheid. Opkomende technologieën voor een betere extractie van context in de toekomst kunnen gebaseerd zijn op thermische infraroodcamera's. Met betrekking tot de ontwikkeling van algoritmen, is het duidelijk dat Deep Learning en convolutionele neurale netwerken (CNN's) al een belangrijke rol spelen om surveillancesystemen intelligenter en robuuster te maken, maar de complexiteitsaspecten en de efficiëntie van deze nieuwe technieken moeten duidelijk verder worden bestudeerd voor het begrenzen van de systeemkosten.

Contents

Summary **i**

Contents **ix**

1 Introduction **1**
1.1 Potential of embedded context-based automatic image and video analysis . . . 1
1.2 Key aspects of embedded context-based automatic surveillance image and video analysis . . . 2
1.2.1 General challenges in context-based image and video analysis . . . 4
1.2.2 Research Scope of the present thesis . . . 4
1.3 Specific problem statement . . . 6
1.4 Contributions . . . 7
1.5 Outline and scientific background . . . 9

2 Technology overview **13**
2.1 Introduction . . . 13
2.1.1 Surveillance objects, background and understanding . . . 13
2.1.2 Different levels of understanding . . . 14
2.2 Types of context information . . . 18
2.2.1 Semantic, spatial, and scale . . . 18
2.2.2 Broader view of context . . . 19
2.3 Pixel-level visual features . . . 20
2.3.1 Spatial domain . . . 20

2.3.2 Transform-based 23
2.3.3 Temporal 25
2.4 Region-level features 26
2.4.1 Salient region detection 26
2.5 Semantic region labeling algorithm steps 28
2.5.1 Step 1: Image segmentation 29
2.5.2 Step 2: Feature extraction 31
2.5.3 Step 3: Classification 32
2.6 Conclusions and discussion 37

3 Fast salient region detection 41
3.1 Introduction 41
3.2 Related work 42
3.3 Salient region detection approaches 43
3.3.1 Edge-based detection 44
3.3.2 Connected components 45
3.3.3 Hough lines 45
3.3.4 Discrete Cosine Transform (DCT) 46
3.4 Experimental results 47
3.4.1 Description of the datasets 48
3.4.2 Settings 48
3.4.3 Evaluation of the results 49
3.5 Conclusions and discussion 52

4 Semantic region labeling 57
4.1 Introduction 57
4.2 Overview of semantic labeling approaches for specific regions . 58
4.3 Semantic labeling of specific regions 60
4.3.1 Sky detection 60
4.3.2 Water detection 61
4.3.3 Road detection 61
4.4 Experimental results for semantic labeling of specific regions . . 62
4.5 Discussion and conclusions for semantic labeling of specific regions 69
4.6 Generic semantic region labeling 71
4.6.1 Stage 1 of generic semantic region labeling: uniform regions 73
4.6.2 Stage 2 of generic semantic region labeling: feature extraction 74
4.6.3 Stage 3 of generic semantic region labeling: classification 75
4.7 Experimental results of generic region labeling 77

4.7.1 Complexity-reduced feature selection in generic region labeling approach . . . 80
4.8 Conclusions and discussion . . . 80

5 **Complexity analysis** **85**
5.1 Introduction . . . 85
5.2 Context-based system . . . 86
5.3 Computational complexity estimation method . . . 88
5.4 Complexity analysis of salient region detection approaches . . . 89
5.4.1 Execution time . . . 89
5.4.2 Complexity estimation of our DCT-based salient region detection . . . 90
5.4.3 Complexity estimation of Rahtu's saliency detector . . . 93
5.5 Complexity analysis of semantic region labeling approaches . . . 94
5.5.1 Complexity estimation of gravity-based semantic region labeling . . . 94
5.5.2 Complexity estimation of Millet's approach . . . 97
5.6 Discussion and conclusions . . . 100

6 **Applications** **105**
6.1 Introduction . . . 105
6.2 Use case 1: Motion context and region labeling for moving ship detection in port surveillance . . . 107
6.3 Use case 2: Automatic traffic sign detection and region labeling for traffic action recognition . . . 112
6.4 Use case 3: Detection of moving cars in traffic surveillance . . . 117
6.4.1 Motion context and region labeling for car detection in traffic surveillance . . . 117
6.5 Motion context and salient regions for detecting moving cars in traffic surveillance . . . 119
6.5.1 Use case 4: Fast abnormal-event detection from video surveillance . . . 124
6.6 Comparison between our gravity-based and a deep learning-based region labeling system . . . 127
6.7 Summary and Conclusions . . . 131
6.7.1 Summary . . . 133
6.7.2 Conclusion and discussion . . . 134

7 **Conclusions** **135**
7.1 Conclusions of the individual chapters . . . 135
7.2 Discussion on research questions . . . 138

7.3 Overall discussion and outlook 141

Complete Bibliography **145**

Acknowledgements **161**

Curriculum Vitae **163**

List of publications **165**

1

Introduction

1.1 Potential of embedded context-based automatic image and video analysis

Visual surveillance in dynamic scenes using video analysis has a wide range of potential applications, such as industrial inspection for manufacturing, disease surveillance in healthcare, security for infrastructures and important buildings, traffic surveillance in cities and highways, detection of military targets, etc.

Video surveillance has been a key component in ensuring security at airports, banks, casinos and correctional institutions. Recently, governmental agencies, businesses and even schools are turning towards smart video surveillance systems as a means to increase public security. The objective of smart video surveillance is to alert police or security officers in case there is any dangerous or suspicious event occurring in a location under video surveillance. Automated video surveillance systems reduces labor costs so less human operators have to observe many video feeds simultaneously. Furthermore, the chance of missing an important event in one of the many surveillance videos among the huge amount of information contained in parallel viewing of video channels is decreased. To substitute human operators by a smart video surveillance system, such system should be able to take decisions autonomously by means of video analysis so as to decide in which situation an alert should be provided in real-time. For reliable decision making, video surveillance systems need to include accurate and reliable video analysis techniques to correctly interpret events in surveillance scenes. Although there have been substantial advances in this field, there is still a need for improving the existing systems in terms of reliability and robustness with respect to event interpretation and a real semantic understanding of scenes.

Adding extra information about objects and/or scenes can lead to a better classification and understanding of events occurring in the surveillance scenes and thereby the reliability and robustness of the systems can be improved. For example, a car detected in a parking place is a normal situation, whereas a car standing on tramway rails is a reason to raise an alarm. In this example, the rails are static object information from the surroundings acting as contextual information. Such extra information leads to a better classification and understanding of events occurring in the surveillance scenes and is referred as contextual information.

Contextual information is defined based on the relationship between a scene and objects/regions within that particular scene. Experiments in scene perception have shown that the human visual system makes extensive use of these relationships for facilitating object detection and scene understanding [1]. Inspired by this concept, the overarching objective of developing context-based automatic surveillance systems is that the systems take the relationships between the scene and the objects (i.e. contextual information) into account. The potential benefit is that a higher level of object detection and scene leading to an improved event understanding. This is a challenging aspect which is not often considered in research and most current approaches are not designed to make use of contextual information.

Context-based image and video analysis algorithms can be embedded as a processing block in another larger algorithm or another application. For example, a context-based region labelling algorithm that is used to semantically label water region, can be embedded and used in a larger ship detection algorithm. Such algorithms or systems are referred to as embedded context-based algorithm or systems.

1.2 Key aspects of embedded context-based automatic surveillance image and video analysis

A typical context-based video surveillance system may comprise several cameras, recording equipment and screens for visualization of the video streams to the security operators. An overview of such a system is shown in Figure 1.1. A brief description of each part is provided as follows:

- *Network and Image/Video Data Storage*. Several cameras are connected to the system via a network so that the generated videos are transferred to the system and may be stored in large databases for later retrieval.

- *Video Analysis*. To enable the automated processing of video streams from surveillance cameras, context-based video analysis algorithms process the video stream in (near) real-time and extract objects and scene-related information. Processing context information can play a useful role in object and region detection by reducing the number of object categories and positions that need to be considered.

- *Metadata*. The output of context-based video analysis algorithms is stored along with the video data and is called metadata: semantic information about the video.

- *Output*. The output of video surveillance systems can be alarming signals which alert security officers in case of alarming situations identified by a context-based decision making system. In addition, information related to the object/regions can be provided to the operator for further analysis of the scenes.

Note that the context-based video analysis algorithms are embedded inside the security system in Figure 1.1. We want to remark that the physical location of these analysis algorithms is not important. They can operate on a central processing server or be embedded inside the cameras, or even distributed. In practice, this will be often a hybrid solution.

An example of a context-based surveillance system is a traffic surveillance analysis system. In such system, traffic signs and zebra-crossing regions

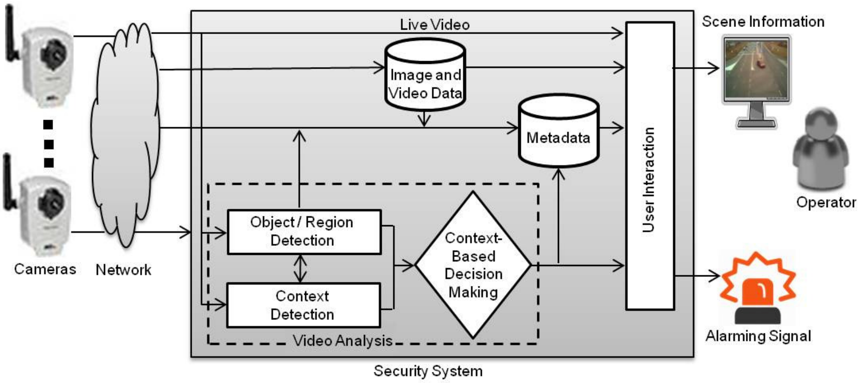


Figure 1.1: Typical video surveillance system setup.

can be detected by the "Context-Based Object / Region Detection" step in Figure 1.1 as contextual information. Additionally, a group of people and possibly other moving objects such as moving cars can be detected. The exploited information can be provided to the decision making step, i.e. "Context-Based Decision Making", in Figure 1.1 to make a decision whether or not an alarming signal should be provided.

1.2.1 General challenges in context-based image and video analysis

Extracting contextual information and exploiting this information to correctly detect regions and interpret actions in surveillance scenes contain multiple challenging aspects. The **first challenge** involves the definition of contextual information, since this can be interpreted in multiple ways. It is evident that context indicates a form of side information that is added to the object of interest. For a car on a crossing, this could be e.g. a traffic sign or the road upon which it is driving. The side information is then of help in assessing the regular or irregular behavior of the car or point to a traffic event. In the thesis, several forms of side or context information will be explored. The **second challenge** involves developing reliable and robust algorithms for extracting the side information from a surveillance video. For example, for a car on a crossing, appropriate algorithms should be used/developed to detect objects i.e. car, traffic signs and road. The **third challenge** involves exploiting the extracted information to improve the semantic understanding of the scenes. For example, the detection of traffic signs near a detected car on a road explains whether or not an illegal action is occurring in the scene. **Finally**, it is challenging to develop generic frameworks which can be scene independent. A generic algorithm should be able to analyze different scenes with different objects e.g. moving cars in roads and moving ships in water. In designing algorithms to address the above-mentioned challenges, it is important that the algorithms are efficiently implemented, so that they can be embedded in (near) real-time systems.

1.2.2 Research Scope of the present thesis

To address the first challenge, i.e. to find out which information is the most informative context information in outdoor surveillance scenes, our research defines the regions based on their saliency and semantic meaning which depends on the application. For example, regions containing man-made objects such as abandoned suitcases in airports can be considered as a region with high saliency. Additionally, it is proposed that regions can be labeled semantically such as water, road and sky, etc., which are the most common regions in outdoor scenes can be defined. Region analysis is applied in two ways: (1) to detect an arbitrary region in an image for general purposes, or (2) to be embedded in another larger algorithm or an application as side or contextual information. In addition to region-level information, low-level features are also explored such as motion context at pixel level, in order to achieve a better scene understanding. For example, for detecting moving ships in harbor surveillance applications, motion features and region information (e.g. water regions) provide contextual information. In this example, information is categorized at the pixel (motion) and region (water) levels.

To address the second challenge, i.e. to choose algorithms for extracting the information, it is desired to create models that are scalable in computational complexity, while maintaining a good recognition performance. In our research our emphasis will be on developing reliable and robust algorithms with low computational complexity. To achieve the robustness, the proposed approaches are designed independently of the scene and in order to achieve low computational complexity simple algorithms are fused into our designs.

To work out the third general challenge, i.e. to exploit the extracted information, semantically meaningful relations between the extracted information are taken into account . For instance, in the example of ship detection a semantically meaningful concept is that moving ships are recognized by salient motions within water regions because logically speaking ship movements are not expected outside of water regions.

To address the last challenge, i.e. developing generic frameworks, this thesis focuses on providing solutions, which are based on common aspects in various surveillance scenes. For example, the proposed solution for moving ship detections can be applied to moving cars in road regions.

Summarizing, it is within the scope of our research to develop reliable and robust algorithms that are generic and have low computational complexity for extracting and exploiting categorized contextual information in surveillance video analysis.

As a technical consequence of our research scope, we account for contextual information in static background (road, sky, etc.) and moving background (water). Additionally, information at multiple layers of regions are taken into account as side information. For example, zebra crossings and traffic signs are accounted for as supplementary, informative layers of road regions. As such in some of the research cases we design algorithms that can be embedded in another application.

1.3 Specific problem statement

The above observations regarding the context-based surveillance scene understanding motivate the problem statement of this thesis, which can be summarized as follows.

It is our objective to develop techniques to provide contextual information derived from surveillance videos and images, using generic solutions or solution architecture that can handle variations in a given scene. The adopted techniques are suited for (near) real-time applications and can be quickly reconfigured and reused for new surveillance-related applications.

To achieve our objective, contextual information should be defined, categorized and embedded in a solution architecture or in frameworks. The algorithms should be analyzed in terms of computational complexity. The performance of the frameworks should be analyzed for real applications. To this end, the specific Research Questions (RQs) for this thesis are formulated as follows.

RQ1 : *Categorization of contextual information*
As explained above, contextual information can be extracted at different levels from scenes. It is, however, not sufficiently explored which levels can be used for extracting contextual information. Therefore, our first question is:

- **RQ1** : *How do we define and categorize contextual information for outdoor surveillance video applications?*

RQ2 : *Detection of features in video surveillance*
Detection and labeling of regions are key aspects in analyzing and understanding of surveillance videos. It is, however, unclear how salient regions and semantically labeled regions can be used to improve surveillance scene understanding. We therefore ask the following questions:

- **RQ2a** : *How can salient regions detection and semantic regions labeling be used for surveillance applications?*

- **RQ2b** : *Which approaches perform better in terms of accuracy for salient region detection and for semantic region labeling?*

RQ3 : *Computational complexity*
For (near) real-time applications, it is important that the developed algorithms have low computational complexity. For the analysis of the computational complexity of the algorithms, we ask the following questions:

- **RQ3a** : *How is the computational complexity of algorithms estimated?*

- **RQ3b** : *Are salient region detection and semantic region labeling methods developed in this thesis feasible for (near) real-time applications with respect to complexity and do they differ from available methods in the literature in terms of computational complexity?*

RQ4 : *Surveillance video applications*
It is important to investigate whether the theoretical concepts and methods developed in this thesis are applicable and make difference in real surveillance applications. Accordingly, we eventually pose questions:

- **RQ4a** : *Are detecting salient regions and labeling of regions with semantically meaningful labels feasible in practical surveillance applications?*
- **RQ4b** : *In which cases and scenarios does the use of contextual information contributes to obtain more reliable and robust surveillance systems in practice?*

1.4 Contributions

In this section we review the contributions of this research for each chapter.

- **Contributions to salient region detection**
  In our research we have explored the salient regions as a support of context information. These regions provide information about surrounding regions of objects. We have analyzed a number of simple, fast salient region detection techniques based on various features, such as sum of edge pixels, number of connected components, number of straight lines found by the Hough transform and the entropy of Discrete Cosine Transform (DCT) coefficients. We have found that the DCT-based technique provides better results compared to the other salient detection techniques. We have shown that the performance of our DCT-based salient region detection technique outperforms with a relevant approach in the literature.
- **Contributions to semantic region labeling**
  We have introduced semantic labeling of specific regions and generic region labeling approaches.
  On the subject of semantic labeling of specific regions, we have presented our research on detecting three specific regions which occur most frequently in an outdoor scene, i.e. sky, water and road. In particular, in our road detection approach, we avoid using color and texture properties and design a novel road detection technique based on two parts:

(1) heat map-based motion analysis and (2) straight line detection.
On the subject of generic region labeling, our major contribution is introducing a generic framework based on spatial context (in our case vertical position information) for labeling regions. We have introduced two models for generic framework: (1) gravity-based and (2) Global Region Statistics (GRS)-based models. Furthermore, we have compared the region-labeling performance of our semantic region labeling approaches with a relevant approach in the literature and we concluded that our gravity-based region labeling is more accurate than the other approaches.

- **Contributions to complexity analysis**
  We have applied complexity analysis method based on counting native DSP operations and memory storage actions with a basic RISC CPU as a reference model. We have analyzed computational complexity of our DCT-based salient region detection and gravity-based semantic region labeling approaches. In our analysis, we have estimated the complexity of the developed algorithms to show that our algorithms are not only offering accurate region analysis, but also execute with low complexity, to support application in (near) real-time and embedded systems. We have compared the computational complexity of our DCT-based salient region detection and gravity-based semantic region labeling approaches with the relevant salient region detection and semantic region labeling approaches in the literature. Our complexity estimation indicates that our salient region detection and semantic region labeling approaches are feasible for (near) real-time applications with respect to complexity.

- **Contributions in surveillance applications**
  We have contributed with several use cases, i.e. moving ship detection in port surveillance, traffic action recognition, moving car detection from traffic surveillance videos and fast abnormal event detection. The contributions of context are very different in these cases. For example, in port surveillance, the detection of the water region as contextual information can help detect ships, because ships can only travel within the water region. In another example in traffic action recognition for safety, the presence of a zebra crossing along with the traffic signs provide contextual information to identify people who cross the road in safe locations. Furthermore, we have compared our gravity-based region labeling with a recently developed region labeling system [2] which is based on emerging deep learning technology. We have

demonstrated that the accuracy of deep learning-based and our gravity-based semantic region labeling approaches are comparable. Although the deep learning approach is more accurate, it is also known to be clearly more complex.

1.5 Outline and scientific background

This section provides an outline of the main chapters of this thesis and the related publication background. A schematic breakdown of the thesis chapter organization is depicted in Figure 1.2. Chapter 2 provides a technical introduction to context categorization and extracting contextual information from different levels of a scene, i.e. pixel, region/object and scene. The following Chapters 3, 4 and 5, present contributions and algorithms to each of these three areas. We have summarized several surveillance applications using one or another form of context in Chapter 6. Finally, conclusions and future work are presented in Chapter 7. The individual chapters are summarized below with their publication history.

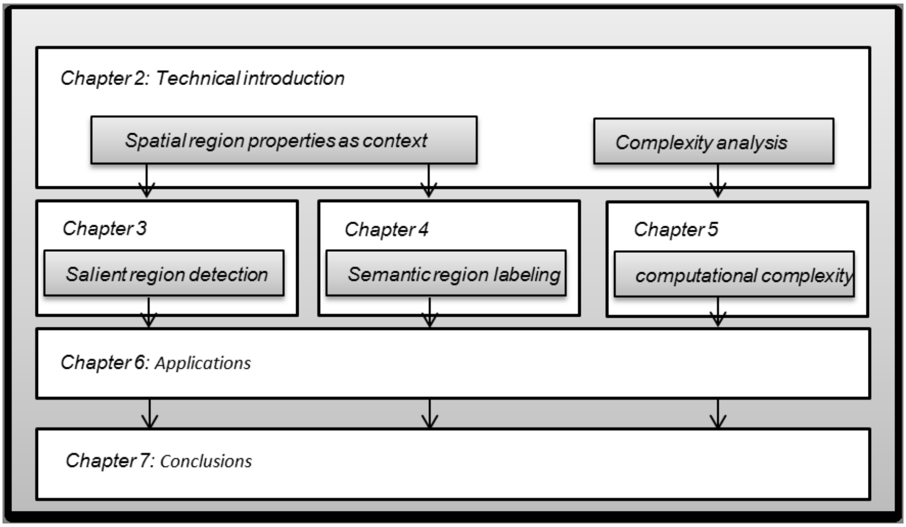


Figure 1.2: Schematic breakdown of the thesis chapters.

Chapter 2. This chapter presents an introductory overview of techniques for the image/video scene understanding. The chapter introduces several levels of scene understanding, i.e., pixel, region/object, context and scene levels. Also, it reviews the state-of-the-art approaches to extract information at the introduced levels.

Chapter 3. This chapter describes our research on scene analysis, namely on detecting salient regions for outdoor surveillance video. We introduce four salient region detection techniques, which are based on sum of edge pixels, number of connected components, the number of straight lines found by the Hough transform and the entropy of DCT coefficients. The experimental results show that our DCT-based approach outperforms a known approach in the literature. The findings of this chapter are published in Int. Symp. Info. Theory Benelux (2014) and IEEE Trans. on CE (2017).

Chapter 4. This chapter addresses our work on research on region labeling for outdoor surveillance applications. Besides our specific region labeling, we introduce the gravity-based and GRS-based models as generic region labeling approaches. Experimental results indicate that our gravity-based model gives the best results and outperforms other similar approaches. The work from this chapter is based on publications in Int. Wksh. Comp. Vision. Appl. (CVA) (2011), Int. Symp. Info. Theory Benelux (2014), Netherlands. Conf. on Comp. Vision. (2014), IEEE Trans. on CE (2017) and Proc. Int. Conf. Consume. Electron. (2017)

Chapter 5. This chapter investigates the complexity analysis of the proposed approaches for DCT-based salient region detection and gravity-based semantic region labeling approaches. Experimental results highlight that both approaches presented and discussed in this study are suitable for real-world applications in surveillance videos. The results of this chapter were used for publication in IEEE Trans. on CE (2017) and Proc. Int. Conf. Consume. Electron. (2017).

Chapter 6. This chapter outlines different surveillance applications that are constructed using the contextual extraction algorithms from the previous chapters. Example applications are the traffic action recognition, the detection of cars from a moving vehicle with a constantly varying background and active ship tracking in a harbor with a PTZ camera. The final application implements abnormal event detection from video surveillance. The applications discussed in this chapter were published in IEEE Trans. on CE (2017), SPIE Journal Electronic Imaging (2015), Book Chapter in Emerg. Res. on Net. Multimed. Comm. Sys. (2015), Proc. Int. Conf. on Img. Proc. Compt. Vis. Patt. Recog. (IPCV) (2012), the IEEE AVSS Conf. (2013) and Netherlands Conf. on Comp. Vision (2014).

Chapter 7. This chapter sums up the most important findings of the thesis and summarizes the obtained results for the different research questions. It is discussed that a smart choice and combination of simple techniques can lead to high-quality results, which even outperform more expensive and complex techniques.

2

Technology overview

2.1 Introduction

Smart surveillance systems should be able to at least detect and track moving objects, classify these objects and interpret their activities. A large number of surveillance systems have been proposed in recent years. These systems still need improvement in terms of reliability and robustness with respect to event interpretation and a real semantic understanding of scenes. Scene understanding methods proposed in the literature are mostly based on considering objects in isolation from the surrounding scene. Some studies have focused on the concept of scene understanding based on contextual information present in the scene. Here, we review various types and levels of the context as well as feature extraction techniques for contextual information, and provide a summary of different types of context into meaningful categories.

2.1.1 Surveillance objects, background and understanding

Improvement in surveillance systems can be realized by adding additional information about objects and/or scenes, so that a better classification and semantic understanding is achieved. This extra information is typically the context of the behavior of objects or specific information captured from the scene. This thesis aims at exploiting contextual information in two ways: (1) to help to better interpret events based on object behavior with higher reliability and robustness, (2) obtaining a higher semantic level of scene understanding by adding contextual information about the scene itself.

Although it is present in scenes, the automated interpretation of the events and associated object detection in a monitored space is typically completely based on object detection and recognition, while the contextual information e.g. about the surroundings of the objects is overlooked. For example, a car detected on a parking place is a normal situation, whereas a car standing on tramway rails is a reason to raise an alarm (see Figure 2.1).

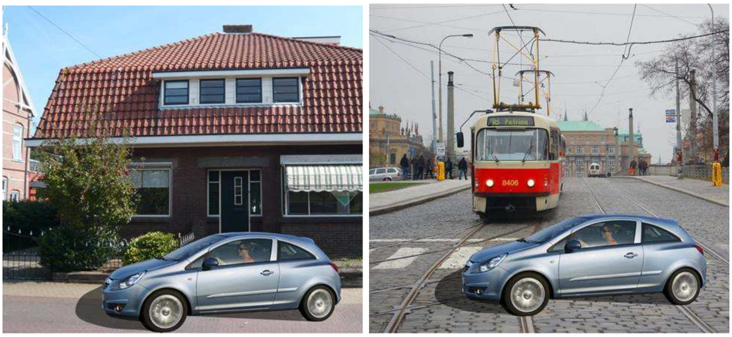


Figure 2.1: Using context to interpret the scene (the key aspect here is the entering location of the car on the public road).

In this example, the rails are static object information from the surroundings acting as contextual information. The benefit is that a higher level of understanding about the crossing car is obtained. In this case, a better understanding of the surrounding elements is achieved in addition to the object detection of the car, jointly leading to an improved scene understanding. When generalizing the previously discussed case for general objects, this discussion results into a schematic diagram as presented in Figure 2.2. The key of this figure is that besides to object detection a structural approach is existing to look to the surrounding information leading to semantic understanding and automatic decision making.

2.1.2 Different levels of understanding

It is important to acknowledge that scene understanding can be applied at several levels. The natural and basic level of scene understanding is at pixel level, such as motion. Just by analyzing the motion, an understanding of the scene is obtained, such as distinguishing between a static background and a moving foreground.

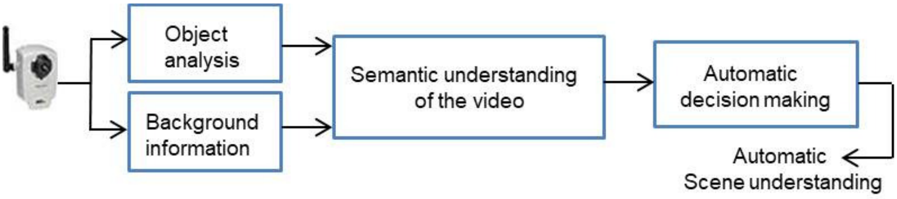

Object analysis
Background information
Semantic understanding of the video
Automatic decision making
Automatic Scene understanding

Figure 2.2: Schematic view of context-based surveillance image and video analysis.

Furthermore, by analyzing different image features, background information can be detected which may lead to a higher level of scene understanding. This background information can consist of regions and/or objects and they are further analyzed to understand the scene. Example of this level of scene understanding is the work of Hazelhoff *et al*. [3]. They proposed an automatic traffic sign detection for improving the traffic scene understanding because a traffic sign supplies important information about the scene. For example, a parked car which is detected together with a forbidden parking sign leads to detecting an illegal situation. Alternative to object detections like traffic signs, we can also perform region analysis of the background, which aims at finding arbitrary or specific regions. For example, a parked car which is detected together with a detected road and building regions lead to the understanding that the car is outside of the public domain. Region analysis for extracting background information is one of the main focus of the areas of this thesis [4].

A higher level of scene understanding can be obtained by including contextual information from the object surroundings in the scene. The work of Creusen and Hazelhoff [5] combined the use of traffic signs at the object level and road markings at the background level, to improve the reliability of the traffic scene understanding. In this thesis the different levels of background information are exploited. This background information can be used in stand-alone fashion such as classifying an input scene by labeling each region. An alternative case with object-of-interest detection where the classified scene and other background information can be used as context for the object detection. Since context is one the key aspects in this thesis to understand surveillance scenes, different levels of contextual information are exploited and will be further discussed.

The highest level of scene understanding can be reached by event analysis which includes not only the objects but also their behaviour. In such a case, by definition we have the objects of interest and our background can contain additional objects and also regions at the same time. This general case may lead to a very intelligent behavioral understanding of the scene. In fact, this highest level can involve several types of objects and also one or more regions to indicate where the specific behaviour is taking place. For example, in order to judge if a crossing group over a street indicates a dangerous situation, we need to detect the street (at region level), the group of people (object of interest) and an approaching a car as our contextual information. Thus, for scene understanding it is not strictly defined how much analysis is needed, as this depends on the application. This level is also further discussed in this thesis [6].

Table 2.1 illustrates different levels of surveillance scene understanding for embedded systems which are considered in this thesis. The table shows the levels of scene understanding, the focus of this thesis based on its involved chapter and an example. In fact, this illustrates that depending on the application and the purpose of the understanding, we need to extract different information. For example, as we discussed above in order to distinguish static background and moving foreground, we need to extract the information at the pixel level which in this case is motion information.

In general, finding several objects or/and regions leads to understanding the actions of the object of interest or the events in the specific area of the scene. Therefore, object or/and region detection approaches are needed. Then, to detect an arbitrary region or object of interest, specific feature information needs to be analyzed. Finally, a learning method may be exploited to detect these objects and identifying behaviour. Figure 2.3 shows the general steps of scene understanding which can be applied in any layer of the scene understanding. For example, to understand group behaviour in a scene, first a moving group should be detected, which can be based on motion or shape properties of the moving blobs. Then, a zebra crossing region as well as traffic sign can be classified to determine if the group is moving legally across the road.

Understanding levels	Description	Chapter of thesis	Examples
Level 1	Pixel	-	Motion
Level 2	Object/Region	3, 4	Labeled regions
Level 3	Context	3, 4	Labeled regions
Level 4	Event	6	Street crossing of a group

Table 2.1: Levels of surveillance scene understanding for embedded systems.

It is important to note that contextual information for scene understanding can be extracted at different levels of surveillance scenes. This notion

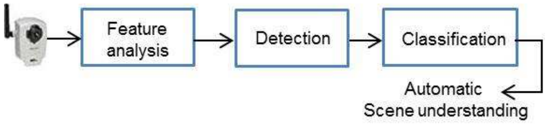


Figure 2.3: Schematic view of finding an object for scene understanding.

leads to the following contextual information types corresponding to the object of interest:

- Feature information at the pixel level, such as color, texture, edge, motion, etc.,
- Properties at region level, such as salient regions and semantically labeled regions,
- Information at the level of scene understanding such as semantically meaningful information, specific objects or behavior, etc.

The scene understanding research is mostly based on considering objects in isolation from the surrounding scene. Examples of publications are the work by Efros *et al.* [7], Minnen *et al.* [8] and Parameswaran and Chellappa [9]. These references are not suitable for our application because they do not take any contextual information into account. While understanding events in a video is certainly interesting, without using contextual information it can be difficult or even impossible to interpret if a situation needs operator attention. Another interesting research topic for surveillance is group tracking. The work of both Lanz [10] and Bazzani *et al.* [11] showed methods of tracking groups of people in surveillance applications. Again we can comment that without contextual information it is difficult to automatically recognize potentially dangerous or suspicious activities. The work of Bao *et al.* [12] introduced a robust approach to detect moving ships by jointly using semantic and motion context. Finally, the work of Marques *et al.* [1] introduced a method that uses contextual information to improve the detection and understanding of the rest of the scene. While certainly interesting, the application in this work is different from ours, because we aim at exploiting contextual information not only to improve the detection and recognition of the scene, but also to interpret human behavior.

The first part of this chapter is dedicated to various types and levels of the context as well as feature extraction techniques for contextual information. Later, we will not concentrate on any specific region as context anymore, but we will focus on any information such as labelled or salient region so that the combination can lead to scene understanding. There are several areas of considerations in this thesis. The area of generic use of context information is considered in the first part of the chapter. The second area is detecting arbitrary regions such as salient and semantic regions. An application for this area can be natural scene understanding. The second area is discussed at the end of this chapter.

The next section will continue with a summary of different types of context into meaningful categories available in the literature. Section 2.3 will review the latest research developments for extracting visual features as contextual information at the pixel-level. Furthermore, Section 2.4 will address the contextual information extraction at the region level, i.e. extraction of salient and semantic labeled region properties. The extraction of contextual information at the third level, i.e. the level of scene, will be described in Chapter 6 of this thesis where the applications are presented. Semantic region labeling approach are further discussed in details in Section 2.5. Section 6.7 concludes the chapter and contains a discussion.

2.2 Types of context information

This section commences with two main types of contextual information that can be exploited in computer vision solutions. These two types are as follows: (1) semantic, spatial, and scale, (2) broader generic view of context. The section will continue with the context information levels, i.e pixel, region and the level of scene understanding.

In this section we present a summary of different approaches to organize the interpretations and types of context into meaningful groups and categories available in the literature. There is no universal agreement on this topic. What follows are some representative examples of classification of types of context and associated contextual modeling techniques.

2.2.1 Semantic, spatial, and scale

Galleguillos and Belongie [13] referred to the following three main types of contextual information that can be exploited in computer vision solutions.

- *Probability* (occurrence): refers to the likelihood of an object being found in some scenes but not in others. From the point of view of modeling, the semantic context of an object can be expressed in terms of its probability of co-occurrence with other objects and its probability of occurrence in certain scenes. For example, a tennis racket is more likely to co-occur with a tennis ball than with a lemon.

- *Position* (spatial): corresponds to the likelihood of finding an object in some positions and not others with respect to other objects in the scene. For example, the sky occurs above the sea and a road is likely to be appearing below a building.
- *Size* (scale): exploits the fact that objects have a limited set of size relations with other objects in the scene. For example, scale context (size) corrects the segment labeled as plant assigning the label of tree, since plants are relatively smaller than trees and the rest of the objects in the scene.

From the computational modeling viewpoint, Galleguillos and Belongie [13] observed that scale context may be the hardest relation to define, because it requires a more detailed information about the objects in the scene, consisting of the identification of at least one other object in the setting as well as the processing of spatial and depth relations between the target object and other object(s). They also claim that semantic context is implicitly present in the other two types of context — spatial context and scale context — although it can be obtained from a wide variety of other sources, such as strongly labeled training data and external knowledge bases.

It is important to acknowledge that for surveillance applications among the above-mentioned types of context, *position* is very important in this thesis because most of its applications lie in the video surveillance domain, which is typically based on natural scenes. We will use the position as a feature to detect an object or region of interest. Afterwards using that object/region of interest, we can improve the level of scene understanding.

2.2.2 Broader view of context

Divvala *et al.* [14] started from the definition of context as "any and all information that may influence the way a scene and the objects within it are perceived". They compile a list of the many different sources of context that have been discussed in the literature, and add some of their own, resulting in a broader and longer list, as follows.

- *Spatial and time-domain information* such as pixel, 2D scene gist and temporal. Image pixels/patches around the region of interest carry useful information [14]. Examples of pixel context include: image segmentation, object boundary extraction, and several object shape/contour models. 2D scene gist refers to models that use global statistics of an image to capture the "gist" of a scene (e.g., [15]). Temporal context captures temporally proximal information, such as time of capture, nearby frames of a video (optical flow), images captured right before/after the given image, or video data from similar scenes [14].

- *Natural information* such as 3D geometry and geography. 3D geometric context corresponds to models that attempt to capture the coarse 3D geometric structure of a scene, or its "surface layout" (e.g., [16]), which can then be used to reason about supporting surfaces, occlusions, and contact points. Geographic context refers to information about the actual location of the image (e.g., GPS coordinates), or more generic information such as terrain type (e.g., tundra, desert, ocean), land use category (e.g., urban, agricultural), elevation, and population density, among others.

- *Scene condition* such as illumination and weather. Illumination captures various parameters of scene illumination, such as sun direction, cloud cover, and shadow contrast [14]. Weather describes meteorological conditions such as current/recent precipitation, temperature, season as well as conditions of fog and haze [14].

The list above includes contextual aspects that can be captured from other sources which reaches far beyond what can be captured from visual input alone.

2.3 Pixel-level visual features

At the pixel level of an image and/or video, we can characterize its content by color, texture, etc. These visual features can be divided into spatial and transformed-based domains and they can also be presented in the temporal domain. These features and methods for their extraction are further discussed in this section.

2.3.1 Spatial domain

The spatial features can be classified into color, edge and texture descriptors. These features and techniques for their extraction are further described here.

A. Color

Color is one of the most straightforward features utilized by humans for visual recognition and discrimination [17]. Colors can be defined in different color spaces. A color space is a specific organization of colors. A number of color spaces have been used in literature such as RGB, LUV, HSV and CIELAB [18]. A number of important color descriptors have been proposed in the literature, including color histogram [19], color moments (CM) [20], color coherence vector (CCV) [21], color correlogram [22], etc. MPEG-7 also standardizes a number of color features including dominant color descriptor (DCD), color layout descriptor (CLD), color structure descriptor (CSD), and scalable color descriptor (SCD) [18]. Table 2.2 provides a summary of different color descriptor methods and their properties.

Color descriptor methods	Advantages	Disadvantages
Histogram	Simple to compute, intuitive	High dimensionality, no spatial info, sensitive to noise
CM	Compact, robust	limited coverage of color range, no spatial info
CCV	Spatial info	High dimensionality, high computational cost
Correlogram	Spatial info	Very high computational cost, sensitive to noise
DCD	Compact, robust, perceptual meaning	Need post-processing for spatial info
CSD	Spatial info	Sensitive to noise, rotation and scale
SCD	Compact, scalable	No spatial info, less accurate if compact

Table 2.2: Overview of color descriptors and a summary of their (dis-) advantages.

B. Edge

Edge is another visual feature at the pixel level. An edge is a boundary or contour at which a significant change occurs in some physical aspect of an image, such as the surface reflectance, illumination or the distances of the visible surfaces from the viewer. Over the past decades, several edge detection techniques have been developed. The Sobel operators are among the most well-known examples of edge detectors. Sobel operator detects edges by calculating partial derivatives in a neighborhood of 3×3 pixels. The Sobel operator is insensitive to noise and it is executed with relatively small aperture compared to other operators, such Laplacian.

Marr and Hildreth [23] proposed the Laplacian of Gaussian (LOG) operator for edge detection. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection. The Laplacian is often applied to an image that has first been smoothed with a Gaussian filter in order to reduce its sensitivity to noise.

Canny presented an optimal edge detector. The Canny operator can give the edge information of both intensity and direction [24]. The detectors mentioned above can be fast but may lead to missing some edges because of operating at the pixel level.

Subpixel-level methods can solve the problem of detection precision. One of the earliest techniques for subpixel edge detection was proposed by Hueckel [25]. He determined edge parameters by fitting image data to a Hibert space of nine parameters. In this technique, a point is declared as an edge point if the computed edge parameter values for that point are sufficiently close to the ideal edge model. A disadvantage of this technique is that it is difficult to detect isolated points [25].

To summarize, it has been shown that under noisy conditions, Canny, LoG and Sobel detectors exhibit better performances compared to other methods. It has been observed that Canny's edge detection algorithm is computationally more expensive compared to LoG and Sobel operators [26].

C. Texture

Texture is another important image feature at the pixel level. While color is usually a pixel-level property, texture can only be measured from a group of pixels. Texture can be defined as spatial arrangements of intensity in image pixels. For example, a tree leaf has a higher degree of texture than sky. Due to their strong discriminative capability, texture features are widely used in semantic learning techniques. Texture has been well studied in image processing and computer vision [27]. A number of techniques have been proposed to extract texture features. Based on the domain from which the texture feature is extracted, they can be broadly classified into spatial texture feature extraction methods and transform-based texture feature extraction methods. In this section, we describe several spatial texture feature extraction methods. Details of transformed-based texture feature extraction methods are addressed in Section 2.3.2. The spatial texture feature extraction techniques can be classified into structural, statistical and model-based approaches [18].

Structural techniques describe textures using a set of texture primitives (textons or texture elements) and their placement rules [28]. In these techniques, textons are organized into a string descriptor, while syntactical pattern recognition techniques are used to find similarity of two descriptors.

Statistical texture feature extraction methods characterize texture as a measure of low-level statistics of grey-level images. The common spatial domain statistical features are moments [28] [29], Tamura texture features [30] [31] and features derived from grey level co-occurrence matrix [29]. Statistical feature extraction methods are compact and robust. However, they are not sufficient to describe the large variety of textures present in surveillance videos.

In model-based techniques, texture is interpreted using stochastic (random) or generative models. Model parameters characterize the underlying texture property of the image. The widely used texture models are Markov random field, simultaneous auto-regressive model, fractal dimension, etc. [18]. Model-based techniques involve optimization to derive model parameters and therefore they are usually computationally expensive [18]. As such, model-based techniques are not suitable for (near) real-time implementation for embedded systems which is one of the main purposes of this thesis. Texture descriptors in transformed domain can be very informative as they contain information on the frequency components of the present textures, and transformed-based descriptors can be extracted in real-time 2.5.2. These transformed-based feature extraction methods are analyzed in the next Section 2.3.2.

2.3.2 Transform-based

Transform-based texture feature extraction techniques involve mapping the image intensity data into a given transform domain [32]. The basic idea of these techniques is to take into account transform coefficients for extracting features. The common transform-based feature extraction techniques include Discrete Cosine Transform (DCT), Fast Fourier Transform (FFT), Gabor filter and Hough Transform. Here, we briefly describe these methods.

DCT can be chosen to extract features from an image because of its high capability of energy compression and availability of fast computational algorithm [33]. The DCT can be applied to the entire image or to sub-image of various sizes. Here, we specify the 2D DCT on each pixel of a sub-image of $N \times N$ pixels by:

$$F(u,v) = \frac{2}{N} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \cos\left[(2x+1)\frac{\pi u}{2N}\right] \cos\left[(2y+1)\frac{\pi v}{2N}\right] f(x,y), \tag{2.1}$$

where $u, v = 0, .., N-1$, $C(u)$ and $C(v)$ are $1/\sqrt{2}$ for $u, v = 0$ and otherwise unity, $f(x,y)$ is the intensity of the pixel in the input image in row x and column y and $F(u,v)$ is the DC coefficient in row u and column v of the DCT matrix [34].

The FFT can alternatively be applied on an image in order to extract image features. The FFT produces a complex-number valued output image which can be displayed with two images, either with the real and imaginary part or with magnitude and phase. More often, only the magnitude of the FFT is exploited in image processing since it contains the most of the information of the geometric structure of the spatial domain [35].

Gabor wavelet decomposition is another transform-based feature extraction technique. In image processing, the most attractive and active application area for Gabor wavelet decomposition filters has been texture segmentation. Besides texture, Gabor filters have been used in edge detection, line segmentation, and shape recognition [36]. A Gabor filter basically analyzes whether there are any specific frequency content in the image in specific directions in a localized region around a pixel or region of analysis. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. Gabor filters are defined by the Gaussian kernel scales (sometimes referred as kernel size) and the orientation of the sinusoidal plane. Fogel and Sagi described the basics of 2D Gabor filter [37]. Despite the success of Gabor feature extraction methods, the costs of Gabor transformations in these methods are also rather high, including both the computational cost and the storage space. For example, for 5-scale and 8-orientation Gabor filters in a single image, 80 convolutions are required to generate the features. The many convolutions make Gabor filter- based feature extraction a complex process, preventing its wide acceptance in practical applications [38]. An alternative to the Gabor filter is the Log-Gabor filter proposed by Field [39]. He suggested that natural images are better coded by filters that have Gaussian transfer functions when viewed on the logarithmic frequency scale. The proposed Log-Gabor filter provides an efficient implementation by constructing a filter in the frequency domain. In this method, the convolution is advantageously replaced by multiplying this frequency domain filter by the FFT of the image and taking the inverse FFT. This leads to a more efficient feature extraction technique. Figure 2.4 depicts the efficient implementation of convolution of Log-Gabor filters in the frequency domain [39].

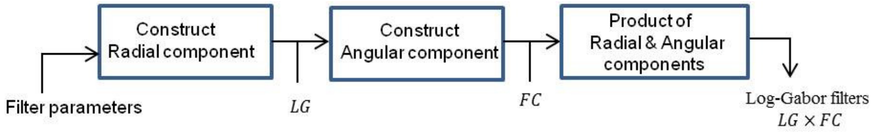

Construct Radial component
Construct Angular component
Product of Radial & Angular components
Filter parameters
LG
FC
Log-Gabor filters
$LG \times FC$

Figure 2.4: Efficient implementation of convolution of Log-Gabor filters in the frequency domain [39] for 1 scale and 8 orientations.

Another transform-based technique to extract features is the Hough transform. The well-known Hough transform is a method for the detection of straight lines in an image. The Hough transform is a line-to-point transformation from the Cartesian space to the polar coordinate space [40]. A line in the Cartesian coordinate space can be described by:

$$\rho = x * \cos\theta + y * \sin\theta, \tag{2.2}$$

where (x, y) denotes a pair from a set of image coordinates, which are lying on a straight line. The transform is quantizing the Hough parameter space into accumulator cells. As the algorithm proceeds, each (x, y) is transformed into a discretized (ρ, θ) curve and the accumulator cells which lie along this curve are incremented. Resulting peaks in the accumulator array represent strong evidence that a corresponding straight line exists in the image [41].

2.3.3 Temporal

When a video of a scene is available temporal features can be extracted to describe content and context of the scene. These features are called temporal features. Motion is a very important visual feature in temporal domain which is discussed in this section.

A. Motion

Motion features are very important in surveillance systems. Motion-based segmentation helps detecting regions corresponding to moving objects such as vehicles and humans. Detecting moving regions provides a focus of attention for performing tracking and behavior analysis because only these regions need to be considered.

Conventional approaches for motion segmentation include block matching, image differencing, background subtraction and optical flow [42]. The performance of the block matching approach is largely affected by the choice of the block size, therefore this approach is not currently common for motion segmentation. Image differencing or background subtraction techniques are often used to find moving blobs in consecutive frames[43][44][45][46]. The drawback of these methods is that they are sensitive to sudden changes in the background, e.g. illumination changes. Optical flow-based methods can detect moving objects even in the presence of camera motion [42]. Therefore, optical flow-based methods are widely used in the surveillance research for extracting moving objects [47][48]. Optical flow-based motion estimation uses characteristics of flow vectors of moving objects over time to detect moving regions in an image sequence, relating each image to the next. Each vector represents the apparent displacement of each pixel from image to image [49]. The Lucas–Kanade method [50] is widely used to implement an optical flow approach. The Lucas-Kanade optical flow algorithm is a simple technique which can provide an estimate of the movement of pixels in successive images of a scene [51].

2.4 Region-level features

This section discusses two main types of approaches for detection of region properties, i.e. salient region detection and semantic region labeling. The detection of region properties can have two general applications: (1) to detect an arbitrary region in an image for general purposes, or (2) to be used as a side or contextual information when it is inserted as a processing block in another larger algorithm or another application. This section is dedicated to salient region detection techniques: (1) feature-based and (2) model-based saliency. The semantic region labeling technique will be introduced in the model-based saliency approach and it is further discussed in more detail in Section 2.5.

2.4.1 Salient region detection

For dynamic scene analysis, automatic detection of informative regions such as salient region is a challenging task for surveillance applications due to the large variations of the scene material. Studies in psychology and cognition have found that, when looking at an image, our visual system would first quickly focus on one or several "interesting" regions of the image prior to further exploring the image contents. These regions are often called salient regions. If such a region is detected, a further advanced video analysis can be applied later exclusively to that region. By providing a salient region to a security officer, without further in-depth analysis, the time for the first-stage visual analysis of the scene is reduced significantly, so that the officer can immediately respond to alarming situations. It can also reduce both bandwidth needed for sending the information and memory needed for storage.

Salient region detection techniques can be divided into two broad families of approaches: feature-based and model-based saliency. In the following subsections, we describe the advantages and disadvantages of each family of salient region detection approaches.

A. Feature-based saliency

In the literature, three spatial features are often used for feature-based detection: color, intensity (or intensity contrast, or luminance contrast) and orientation. Intensity is usually implemented as the average of three color channels.

Orientation is obtained by a convolution with directional Gabor filters or by the application of oriented masks [52]. The features can also be combined. This, however, does not lead to better saliency detection and it only increases the complexity of the approach.

The motion in video introduces another important feature for the human attention system. The most common technique for motion-based saliency is using optical flow. Optical flow-based motion detection technique was discussed in Section 2.3.3 - A.

Features can be found not only in the spatial and temporal domains, but also in the frequency domain. Most common frequency-domain features such as DCT were addressed in Section 2.3.2.

The feature-based approach is scene dependent. Only the visual input is of importance. Other inputs such as tasks or object indicators are of no concern in this approach. Feature-based techniques are fast and therefore suitable for surveillance applications. In this thesis, we will apply different feature-based approaches in Chapter 3 and evaluate their influence on the performance of the salient region detection approaches.

In salient region detection, it is important to decide which feature should be chosen. Figure 2.5 is an example which demonstrates an image of a leaf on a high textured fence. In this example, the fence is detected when taking the texture feature into account. However, the leaf can also be detected if the color is taken into account. This example indicates that the choice of the most relevant features for saliency detection is difficult and scene-dependent. However, a different example can be found in the security application area. If a person with dark clothes is positioned in shadows or in a dark area, he will not be noticed with most visual feature extraction techniques, because this person does not stand out with respect to its surroundings. Therefore, considering only visual features may not help to detect informative regions for security applications. In such cases, high-level features such as a region and/or object should be taken into account in model-based saliency approaches, which are described below.

B. Model-based saliency

Model-based saliency approaches are top-down methods which are determined by high-level features like objects and/or regions [52]. These methods involve training in order to learn the appearances of a specific salient region before the system can look for it. The basis of many salient detection methods dates back to Treisman and Gelade's [53] "Feature Integration Theory," where

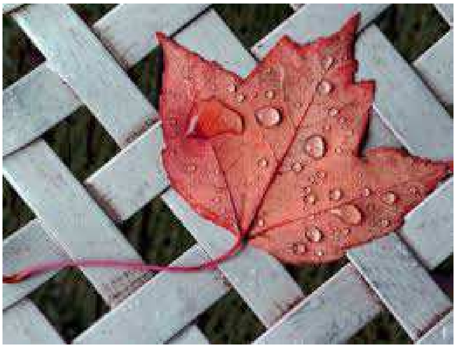


Figure 2.5: Leaf on a fence. It is not evident which feature should be chosen for saliency detection.

they stated which visual features are important and how they are combined to direct human attention. Koch and Ullman [54] then proposed a feed-forward model to combine these features and introduced the concept of a saliency map, which is a topographic map that represents conspicuousness of scene locations. They also introduced a neural network that selects the most salient location and employs a mechanism to allow the focus of attention to shift to the next most salient location [52].

One attractive application for model-based saliency is to obtain labeled regions with semantically meaningful labels (e.g., road, sky, etc.). These methods can be advantageously used to classify pictures in a large database. In another application, these methods can be exploited to improve the analysis of events or contribute to more reliable object detection for surveillance applications. In fact, saliency detection is an important area which can be used as a side-processing block in another larger algorithm or an application. For example, a salient region detection algorithm can be inserted in a certain application that is called semantic region labeling algorithm. Semantic region labeling is an important area that we will consider in this thesis. Therefore, in the following section, we review the details of common steps for region labeling algorithms.

2.5 Semantic region labeling algorithm steps

Existing approaches in region labeling often produce a set of textual semantic labels as "words", describing the image content without linking these words to particular segments of the image. One of the challenging aspects of automatically labelling image regions is to take into account the contextual information which is present in a scene. The annotation of regions ignoring their context and focusing only on the information within the object boundaries (such as color and texture information) is often an impossible task [55]. Kluckner *et al.* [56] proposed a labeling approach by integrating additional contextual constraints, such as class co-occurrences into a randomized forest classification framework. Ladicky *et al.* [57] incorporate object co-occurrence in Conditional Random Fields (CRF). The co-occurrence model, however, tends to require large numbers of training samples to estimate the correct probability.

The first step in semantic region labeling is to efficiently extract descriptive visual features from an image. The features can be extracted at pixel or at region level. Region-based feature extraction needs prior image segmentation, while pixel-based features are directly extracted from each image pixel independently. Although extraction of features at both pixel and region levels is used in the existing region labeling techniques, the trend is towards extracting features at the region level. Region-based labeling techniques involve three steps: image segmentation, feature extraction and classification [18]. In this section, we address common region-based labeling algorithm steps.

2.5.1 Step 1: Image segmentation

Image segmentation is usually the first step to extract region-based image representation. The segmentation algorithm divides the image into different components based on feature homogeneity. A number of segmentation approaches is described in the literature, where theses approaches are based on grids, clustering, contours, graph, and region growing methods. For a comprehensive segmentation review, readers are referred to [58].

Because automatic image segmentation is a difficult task, many techniques simplify this task using grid-based approaches to roughly segment images into blocks [58]. Visual features are then extracted from these blocks. The advantage of a block-based approach is the low computational costs. However, this simple technique does not describe the semantic components in images well.

Clustering algorithms, like K-means, are used to cluster pixels into different groups [58], with each group identifying a region. In most cases, an image is first partitioned into blocks of size 4×4 pixels. Color and/or texture features are extracted for each block. Then, K-means is applied to cluster the block's feature vectors. A region is formed with the pixels belonging to blocks of the same cluster. The major issue with this approach is that it needs to predefine the number of segments based on heuristics. An inappropriate choice of the number of clusters yields poor results. The other issue is that the algorithm assumes data are in spherical clusters, so that the mean values are near the cluster centers. This assumption, however, is usually not valid.

Another segmentation approach is contour-based segmentation. The main idea of contour-based segmentation is to evolve a curve around an object. The evolution stops when the curve coincides with the boundary of an object. Unlike the cluster-based segmentation algorithm, contour-based segmentation algorithms do not need the prior assumption for the number of clusters [68–70].

An alternative approach is graph-based segmentation technique. Shi and Malik [59] proposed a graph-based segmentation algorithm known as normalized cut (NCut). The NCut method represents an image as a graph where vertices are image pixels and the edge weights represent the feature similarities between pixels. Image segmentation then becomes a graph-partitioning problem (see Figure 2.6). The idea is to partition the vertices of the graph into disjoint sets so that the total similarity between different sets is minimized. Each set is regarded as region. As the number of pixels in an image is large, there are exponential numbers of possible partitions of the graph. As a result, it is computationally expensive to find the optimal partition. Tao *et al.* [77] improves the NCut by pre-segmenting images using the mean shift algorithm. Instead of using pixels, the regions of the initial segmentation are used as vertices in the NCut algorithm. Hence, the computational cost is reduced, and the performance is more robust. The basic NCut exploited on color features only. Malik *et al.*dey2010review] extend it to incorporate texture features. Another important concept in graph theory is pixel connectivity. The notation of pixel connectivity describes a relation between two or more pixels. For two pixels to be connected, they should fulfill certain conditions on the pixel brightness and spatial adjacency. First, in order for two pixels to be considered connected, their pixel values must both be from the same set of values V. For a grayscale image, V might be any range of grayscale, e.g. $V = 22, 23, ...40$. To formulate the adjacency criterion for connectivity, first the notation of neighborhood is introduced. For a pixel p with the coordinates (x, y) the set of 4 neighboring pixels is given by:

$$N_4(p) = \{(x+1, y), (x-1, y), (x, y+1), (x, y-1)\}\,, \tag{2.3}$$

An alternative is to consider 8 neighboring pixels. In this case the set is defined as:

$$N_8(p) = N_4(p) \cup (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1), \tag{2.4}$$

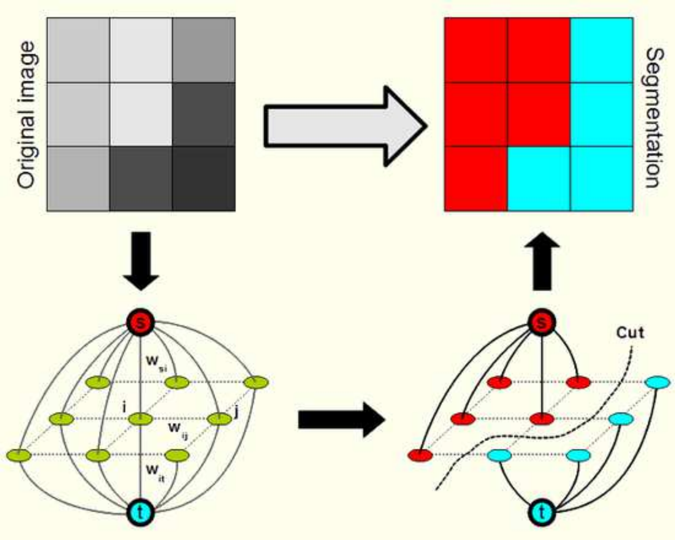


Figure 2.6: Establishing a flow network from a 2D image as required by graph-cuts [22].

An alternative approach for segmentation is region growing. This approach groups pixels or smaller regions into larger regions. At first, pixel colors of the image are quantized into a number of classes. Then, pixels in the image are replaced with the color-class labels. A class map is subsequently formed and region growing is followed on the class map. Pixels with more homogeneous neighbors are assumed to be interior pixels of possible regions. These pixels are selected as candidate seed points and regions are grown around these seed areas. As this method looks for both color and texture homogeneity, the segmented regions have highly homogeneous characteristics [18]. When the segmentation is performed, the following step for region labeling is the feature extraction step.

2.5.2 Step 2: Feature extraction

Color and texture features carry descriptive information on the image content. Various feature extraction techniques are discussed in Section 2.3.

Color may be described in an intuitive way using the HSV (or HSL, or HSB) space, which is widely used in computer graphics. The three color components are hue, saturation and value (or lightness, brightness). The hue is invariant to the changes in illumination and camera direction and hence more suited for object retrieval. RGB coordinates can be easily translated to the HSV (or HSL, or HSB) coordinates by a simple formula [29].

Texture is another important property of images. Various texture extraction techniques are discussed in Section 2.3, among which log-Gabor filters, as proposed by Field [39], which are most robust. Log-Gabor provides an efficient implementation of convolution in the frequency domain.

In this thesis, the HSV color space together with a group of log-Gabor filters and spatial information in the vertical direction are extracted for this step.

2.5.3 Step 3: Classification

Once images are represented with low-level features, the features are fed directly into a classification method to learn from image samples. Depending on the whether there is prior knowledge of the classes, the classification methods are divided into two groups, supervised and unsupervised classification techniques. Once the classifier is trained, the algorithm can be used to annotate new image samples. There are generally three types of supervised annotation approaches. The first approach is the single-labeling annotation using conventional classification methods. The second approach is the multi-labeling annotation, which annotates an image with multiple concepts using the Bayesian methods. The third approach is the web-based image annotation which uses metadata to annotate images [18].

In the following, we discuss the single-labeling annotation using conventional classification methods in detail because it is most relevant to this thesis.

In the single-labeling approach, low-level features are extracted from image content, and the features are fed directly into a conventional binary classifier which gives a “yes” or “no” vote for each particular region in an image. The common single-labeling approaches include [18]: (1) Deep Learning in Neural Networks (NNs), (2) Decision Tree (DT) and (3) Support Vector Machines (SVM). We briefly explain each of them in the following sections.

A. Deep Learning in Neural Networks

A standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations. Input neurons get activated through sensors perceiving the environment, other neurons get activated through weighted connections from previously active neurons. Some neurons may influence the environment by triggering actions. Learning, or credit assignment, is about finding weights that make the NN exhibit desired behavior [60]. Figure 2.7 shows an example of how an example of a neural network that classifies an image region into one of the three broad categories such as sky, water, and earth. Depending on the problem and how the neurons are connected, such behavior may require long causal chains of computational stages, where each stage transforms (often in a non-linear way) the aggregate activation of the network. Deep Learning is about accurately assigning credit across many such stages. An efficient gradient descent method for training NNs called backpropagation was developed in the 1960s and 1970s, and applied to NNs in 1981. Backpropagation-based training of deep NNs with many layers, however, was found to be difficult in practice. In fact, since 2009, supervised deep NNs have won many official international pattern recognition competitions, achieving the first superhuman visual pattern recognition results in limited domains [60]. Recent CNN models solve semantic segmentation jointly, and removes the suboptimal separation in multiple stages.

In the past years, multiple network architectures have been proposed each optimized for its accuracy, speed or size. Important architectures for pixel-to-pixel classification include Resnet [61], Unet [62], Segnet [63], fully convolutional networks [64] and Deeplab [65]. Some networks have been specifically designed to minimize memory usage and inference time, while maintaining accuracy. Such networks are VGG16 [66], Mobilenet [67], Darknet [68] and Squeezenet [69]. Unfortunately, computational complexity and execution time of these networks, are generally not discussed in detail. It is noted that, with growing hardware performance in GPUs and tailoring such computing cores to convolutional neural network (CNN) structure, the required computation power for a neural net is becoming more broadly available. These developments enable that CNN-based solutions are gradually growing into the embedded system area. Although computational complexity of deep NNs has become more feasible, NNs also require massive amounts of data to train the classifier. Most of the aforementioned issues have improved significantly in past 3-4 years, when our research of this thesis was already completed. However, for the sake of completeness, we will make a performance comparison at the end of the thesis in Chapter 6.

B. Decision Tree and Random Forest

A decision tree is a multi-stage decision making, or classification, tool. Depending on the number of decisions made at each internal node of the tree, a decision tree can be called binary or n-ary tree. Different from other classification models whose input–output relationships are difficult to describe, the input–output relationship in a decision tree can be expressed using understandable rules, e.g., if–then rules [18]. Figure 2.8 shows this process.

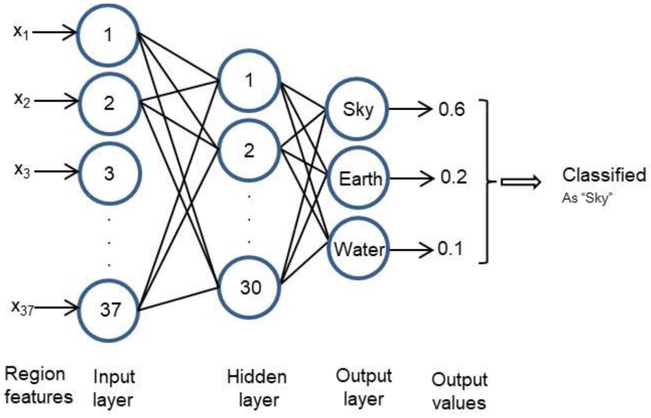


Figure 2.7: Classifying a region using NN [18].

Combining the ideas of decision trees and ensemble methods gave rise to decision forests, that is, ensembles of randomly trained decision trees. The idea of constructing and using ensembles of trees with randomly generated node tests, i.e., random forest, was introduced for the first time in the work of Amit and Geman [70] [71] for handwritten digit recognition. In that work, the authors also propose using the mean of the tree probabilities as output of the tree ensemble.

C. Support vector machines

SVM is a supervised classifier. It has been shown that SVM has high effectiveness in high-dimensional data classifications, even when the training dataset is small [58]. SVM can classify both linear and non-linear data due to the use of kernel mapping. The advantage of SVM over other classifiers is that it achieves optimal class boundaries by finding the maximum distance between classes. It has been successfully applied to a number of classification problems, such as text classification, object recognition and image annotation [58]].

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types such as: (1) Polynomial kernel which is popular in image processing, (2) Gaussian radial basis func-

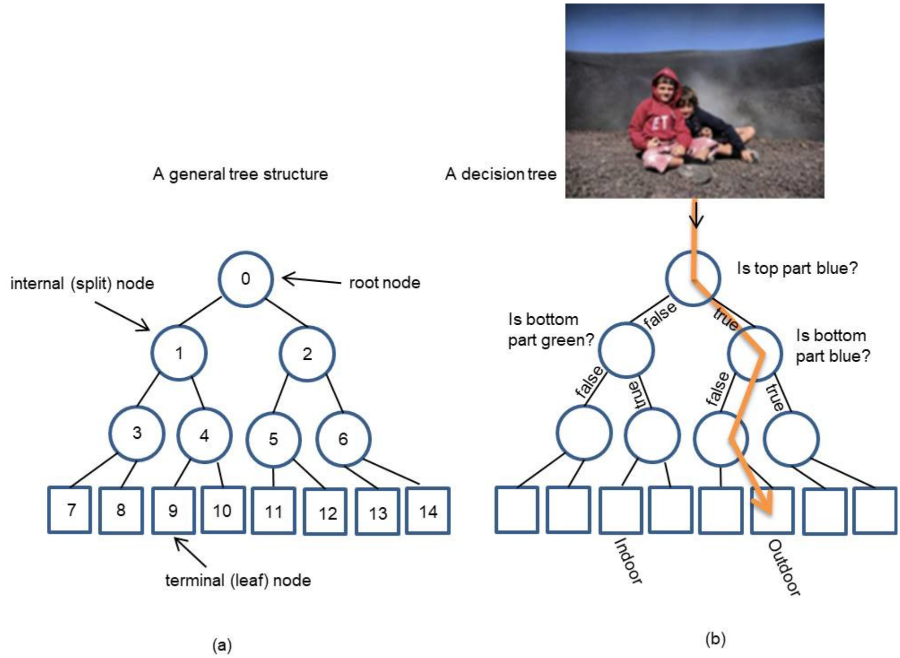


Figure 2.8: Decision tree. (a) A tree is a set of nodes and edges organized in a hierarchical fashion. A tree is a graph with no loops, where internal nodes are denoted with circles and terminal nodes with squares. (b) A decision tree is a tree where each internal node stores a split (or test) function to be applied to the incoming data. Each leaf stores the final answer (predictor). An illustrative decision tree used to figure out whether a photo represents an indoor or outdoor scene [72].

tion (RBF) is a general-purpose kernel; used when there is no prior knowledge about the data, and (3) Linear splines kernel in one-dimension is useful when dealing with large sparse data vectors. It is often used in text categorization. The splines kernel also performs well in regression problems. In this thesis, we use RBF. RBF's equation and further details of SVM approach can be found in Chapter 5 and [73].

An SVM classifier works by finding a hyperplane from a training set of samples to separate them. Each training sample is represented with a feature vector and a class label. The hyperplane is learned in such a way that it can separate the largest portion of samples of the same class from all other samples. An SVM is a binary classifier. However, automatic image classification and annotation needs a multiclass classifier. The most common approach is to train a separate SVM for each concept with each SVM generating a probability value. During the testing phase, the decisions from all classifiers are fused to get the final class label of a test image. Figure 2.9 shows this process. As such, the complete classifier is a two-level process. The base level consists of multiple binary classifiers and the second level fuses the decisions from the base level classifiers.

Chapelle *et al*. [74] applied the above-mentioned basic framework to train 14 SVM classifiers for 14 image-level concepts. In their approach, images are represented with 4096-dimensional HSV histograms. To train an SVM for a particular concept, training images belonging to that concept are regarded as positive samples while the others are regarded as negative samples. Therefore, each trained classifier can be regarded as a "one-versus-all" classifier. During testing, each classifier generates a probabilistic decision. The class with maximum probability is selected as the concept of the test image. Figure 2.9 illustrates multiple SVM ("one vs. all") classifiers. Each SVM independently classifies an input image, while the final decision is fused from the decisions of all SVMs.

Most of the findings show that there is empirical evidence to support the theoretical formulation and motivation behind SVMs. The most important characteristic is the ability of SVM to generalize well from a limited amount and/or quality of training data. Compared to alternative methods such as NNs, SVMs can yield a comparable accuracy using a much smaller training sample size. This is in line with the "support vector" concept that relies only on a few data points to define the classifier's hyperplane [75]. Watanachaturaporn *et al*. [76] found that SVM methods outperformed NNs and decision trees. Therefore, in this thesis multiple SVM ("one vs. all") is applied to assign multiple semantic labels to image segments.

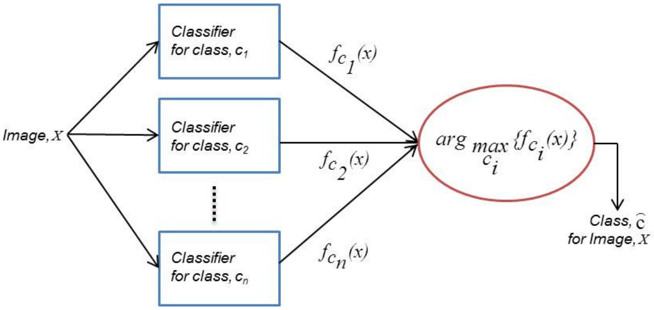

Image, X
Classifier for class, c_1
Classifier for class, c_2
Classifier for class, c_n
$fc_1(x)$
$fc_2(x)$
$fc_n(x)$
$\arg\max_{c_i} \{fc_i(x)\}$
Class, $\hat{c}$ for Image, X

Figure 2.9: Multi-class classifier using multiple binary SVM classifiers.

2.6 Conclusions and discussion

In this chapter we have presented an introductory overview of techniques for the image/video scene understanding. We have introduced several levels of scene understanding, i.e., pixel, region/object, context and scene levels. It was discussed that contextual information can also be exploited at different levels:

- pixel-level information such as color, texture, edge, motion, DCT and Hough-based transform,
- region-level, such as salient and semantic-labeled regions as background information,
- at the level of scene understanding such as semantically meaningful information for specific objects or behavior, etc.

We have reviewed the state-of-the-art approaches to extract pixel-level information in spatial, transform-based and temporal domains. Regarding region-level information, we divide salient region detection techniques into two families of approaches: feature-based and model-based saliency detection approaches. We consider semantic region labeling approach as a form of model-based saliency detection. We have reviewed common region-based labeling algorithm steps, i.e. image segmentation, feature extraction and classification.

At pixel-level we will consider color, texture, edge, motion, DCT and Hough-based transform since they are most common features in spatial, temporal and transform-based domains. At region-level we will consider methods to detect salient and semantically labelled regions. These methods can provide a platform for fast retrieval of important regions in (near) real-time for embedded surveillance systems.

In Chapter 3 we will consider a number of simple, fast saliency detection techniques based on various features, such as the entropy of DCT coefficients, sum of the edge pixels, number of connected components and the number of straight lines found by the Hough transform. Our hypothesis is that, in surveillance application, by providing a salient region, the time for the first-stage visual analysis of the scene can be reduced significantly, so that the officer can immediately respond to alarming situations.

The region labeling approach will be further addressed in Chapter 4. We will develop a general framework for performing automatic semantic labeling of video scenes by combining the local features and spatial contextual cues.

Another challenging aspect of automated surveillance scene understanding systems is to analyze the computational complexity of such systems. In Chapter 5 we will discuss a known metric for estimating the computational complexity based on Mega Operations per Frame (MOPF) or per second (MOPS) [77]. This metric is based on counting native Digital signal processing (DSP) operations, like multiplications, additions, data storing and loading.

Regarding the scene-level contextual information, several traffic surveillance use cases will be addressed later in Chapter 6. We will illustrate four traffic surveillance use cases which are briefly summarized below.

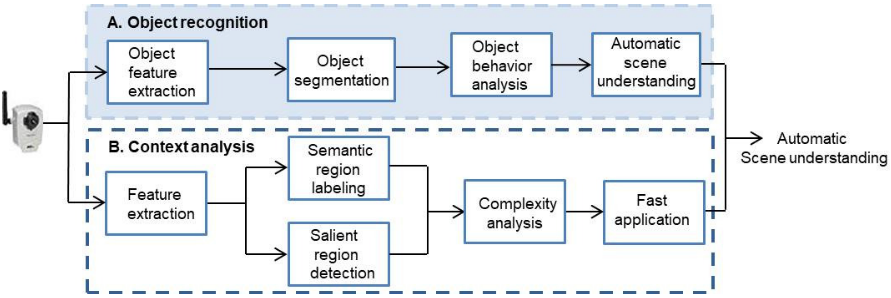


Figure 2.10: Schematic representation of the steps needed for developing a surveillance video analysis system using context information. Such system should have high accuracy and low computational complexity.

- Moving ship detection in harbor surveillance, the semantic region labeling is combined with motion information.
- Traffic action recognition based on semantic region labeling where also automatic traffic sign information is exploited.
- Moving car detection in traffic surveillance, the semantic region labeling and salient region detection are both applied and combined with motion information.
- Fast abnormal event detection which is a novel block-based approach to detect abnormal situations by analyzing the pixel-wise motion context, as an alternative for the conventional object-based approach.

The above contextual information types and levels give a structured overview of the fundamental elements that are used in this thesis. However, the list is not exhaustive, but merely acts as a framework for the design of the video analysis systems. In each chapter, specific additional aspects will come to the foreground that will be addressed in that chapter.

3

Fast salient region detection

3.1 Introduction

Chapter 2 has described several algorithmic components for the construction of surveillance image and video analysis applications. It presented an introductory overview of techniques for the image/video scene understanding. To this end, we have introduced several levels of scene understanding, i.e., pixel, region/object, context and scene. We have also identified three different levels at which context information can be obtained: at the pixel, region, and scene levels. It was discussed that region analysis can be applied in two ways: (1) to detect an arbitrary region in an image for general purposes, or (2) to be used as a side or contextual information when it is inserted as a processing block in another larger algorithm or an application. This chapter is dedicated to our first approach for extracting information at the region level, i.e. salient region detection approach. The second approach, i.e. semantic region labeling will be discussed in Chapter 4.

Studies in psychology and cognition have found that, when looking at an image, our visual system will first quickly focus on one or several "interesting" regions of the image prior to further exploring the image contents. These regions are often called salient regions. For selecting salient regions, several different methods were reviewed in Chapter 2. As opposed to the feature-based saliency detection algorithms which are fast and driven by low-level features, model-based saliency detection algorithms are slow and task-driven. This chapter focuses on feature-based saliency detection which aims at answering at the following question: how can we accurately detect salient regions without any background information [78]. We concentrate on simple and fast methods suitable for real-time applications. It should be noted that the majority of the saliency detection approaches in literature result in saliency maps which are widely employed for unsupervised object segmentation. In our surveillance application, instead of providing saliency maps, where each pixel is assigned its saliency level, we aim at providing the segmentation of salient regions in an input image. This segmentation results in a binary image where the salient region pixels have one value, and the rest has another. By providing the segmented salient regions to a security officer, without further in-depth analysis, the time for the first-stage visual analysis of the scene is reduced significantly, so that the officer can faster respond to alarming situations. It can also reduce both bandwidth needed for sending the information and memory needed for storage.

This chapter is organized as follows. Section 3.2 will review common salient region detection techniques. In Section 3.3 we will introduce and compare a number of simple, fast detection techniques based on various features, i.e., sum of edge pixels using a Sobel edge detector, number of connected components based on the intensity values of neighboring pixels, the number of straight lines found by the Hough transform and the entropy of the frequency-domain features of a DCT. Section 3.4 presents our results. Conclusions and discussion are provided in Section 3.5.

3.2 Related work

In this section we focus on relevant literature targeting fast feature-based saliency region detection methods. Saliency of a region mainly depends on the contrast between an object or region and its surroundings. As such, feature-based saliency region detection methods utilize low-level processing [79]. Low-level saliency methods use local contrast of image regions with their surroundings using one or more of the features of color, intensity, and orientation in the spatial domain. Also the features in temporal and spectral domains can been applied.

Itti *et al.* [80] introduced a saliency model which was biologically inspired. Specifically, they proposed the use of a set of features which are intensity, color, and orientation in spatial domain. The features were normalized and then linearly combined to generate the overall saliency map. In addition to the spatial domain, Itti *et al.* [80] also utilized motion information in the temporal domain in their model. Even though the proposed model was shown to be successful, still many parameters have to be tuned manually [80].

An alternative approach for detecting saliency is based on line finders [81]. The Hough transform is known as an algorithm which can extract lines effectively in the spectral domain [4]. However, this transform generally requires a large storage and a high computational capacity [82].

Feature-based saliency region detection methods can also use the contrast of image regions with respect to the entire image. Zhai and Shah [83] define pixel-level saliency based on a pixel's contrast to all other pixels. However, for efficiency they use only luminance information, thereby ignoring important clues in other channels [79].

In an alternative approach several models were proposed based on information in frequency domain. Recently, a simple algorithm, called the Spectrum Residual (SR), was proposed based on the Fourier Transform [84]. It was argued that the spectrum residual corresponds to image saliency. In another work, the Phase spectrum of the Fourier Transform (PFT) was introduced, which achieved nearly the same performance as the SR [84]. Based on PFT, PQFT [84] was also proposed by combining more features and using the quaternion Fourier Transform. These approaches fail to detect some salient pixels, since they do not incorporate local saliency [84] [85].

Although the techniques described above are interesting because of their algorithmic robustness, they require high computational cost because of their algorithmic complexity. In this thesis, we aim at simple and fast methods suitable for real-time operation as required by surveillance applications. In the next section we will present our salient region detection approaches.

3.3 Salient region detection approaches

In this section we present four detection techniques based on various features: (1) sum of edge pixels using a Sobel edge detector, (2) number of connected components based on the intensity values of neighboring pixels, (3) number of straight lines found by the Hough transform and (4) the entropy of the frequency-domain features of a DCT. We also discuss the parameter choices for each technique. Our hypothesis is that regions including human-made objects such as cars or suitcases are potentially more suspicious. Such regions include high amount of lines, edges or any other high-frequency information.

Figure 3.1 illustrates a schematic view of our salient detection techniques. In the first three approaches, the pre-processing stage transforms an input frame from an RGB format to a gray-scale format. After this, an edge detector is applied. The pre-processing stage for the DCT-based approach consists of extracting the color channels from the input video frame because further processing will be performed per channel. The second block in the flow chart of Figure 3.1 refers to finding salient regions based on various features depending on the corresponding approach. Finally, the post-processing step consists of a temporal filtering to make sure that the salient regions are not switching on and off between consecutive frames, so that flickering in the video output is prevented. The applied temporal filtering method considers a sliding 10-frame interval. If the region appears salient for six out of ten frames, then it becomes salient region for all ten frames. It should be noted that while the first stage is applied to the whole image, the second and third stages are block-based processing stages.

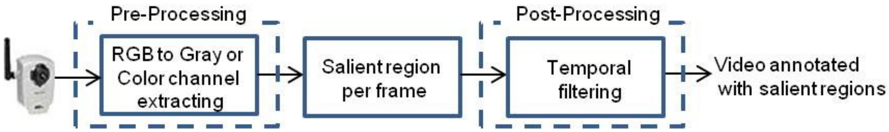

Pre-Processing
RGB to Gray or Color channel extracting
Salient region per frame
Post-Processing
Temporal filtering
Video annotated with salient regions

Figure 3.1: Schematic view of our salient region detection techniques.

3.3.1 Edge-based detection

This approach is based on local-contrast methods for a gray-level image. In this approach, the sum of edge pixels is used to detect a salient region. Edge detection is based on the Sobel edge detector, since it has been shown that under noisy conditions Sobel exhibits promising performance and is computationally inexpensive [79].

The Sobel operator can detect edges by calculating partial derivatives in a 3×3 neighborhood. The details about this algorithm are presented in [86]. In this approach after smoothing an image by a Gaussian filter to reduce noise, the Sobel operator calculates the partial derivatives. The partial derivatives S_x, S_y in horizontal, x, and vertical, y, directions are given as [86]:

$$\begin{aligned} S_x = f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1) - \\ f(x-1,y-1) + 2f(x-1,y) + f(x-1,y+1), \end{aligned} \tag{3.1}$$

$$\begin{aligned} S_y = f(x-1,y+1) + 2f(x,y+1) + f(x+1,y+1) - \\ f(x-1,y-1) + 2f(x,y-1) + f(x+1,y-1), \end{aligned} \tag{3.2}$$

where $f(x,y)$ is the intensity of the pixel in the input channel in row x and column y. Then, the gradient of each pixel is $g(x,y) = \sqrt{(S_x)^2 + (S_y)^2}$. Subsequently, a threshold value t_{grad} is selected. If $g(x,y) > t_{grad}$, the current pixel is regarded as an edge pixel This thresholding results in a binarized image.

Let us describe our method in more detail. After the pre-processing stage, consisting of RGB to gray-scale transform, a Sobel edge detector is applied on an input frame. Then, we divide each resulting frame into number of blocks. For deciding whether a block is salient or not, the algorithm compares the number of edge pixels in each block with a threshold. If this number is above the threshold (T_B), the current block is considered as a salient region. The threshold T_B is calculated empirically by:

$$T_B = (1/(0.8 \times N_{\text{blocks}})) \times N_{\text{edg}}, \tag{3.3}$$

where N_{blocks} is the number of blocks in the current frame and N_{edg} is the total number of edge pixels in the frame.
The major disadvantage of this method is that a region including small-sized edges, may be selected as a candidate for a salient region. For example, fine foliage trees which include a lot of small edges are not good candidates for salient regions. This issue can be solved using the connected components algorithm, which will be discussed bellow.

3.3.2 Connected components

In the connected components approach, we compute the number of connected components to find a salient region in an image. After the pre-processing stage (Figure 3.1), we apply a Sobel edge detector on an input frame. In this approach, we use a 8-connected neighborhood (see Figure 3.2) to find connected components in the resulting frame. The connected components are found based on the flood field approach [87]. Finally, we obtain a labeled map. Each group of connected components has its own label. For each connected component, if the number of pixels it contains is lower than a predetermined threshold, this component is discarded. This allows to avoid small edges that cannot be removed in the sum of edge pixels approach presented in Section 3.3.1. Then, we divide the frame into blocks. For each block, the algorithm checks whether it contains any labels or not. The block is considered as a salient region if it contains at least one label. This approach is more reliable compared to using only an edge detector, as it avoids small connected components that are mostly irrelevant for semantic analysis.

3.3.3 Hough lines

To detect straight lines in an image, we first apply the 3×3 Sobel edge detector. Then, we divide a frame into number of blocks. Later, we apply the Hough transform on each block. For deciding whether each block is salient

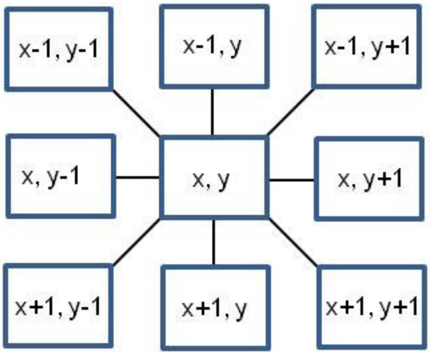


Figure 3.2: 8-Connected neighborhood of a pixel. Variables x, y indicate the row and column of center pixel in a 3×3 neighborhood.

region or not, the number of straight lines for each block is compared to a predetermined threshold. If this number exceeds the threshold, the block is considered as a salient region.

The Hough-based technique has some inherent limitations such as large computing time, massive memory requirements and its incapability in preserving edge pixel connectivity [88]. Therefore, in the following section we will employ a DCT-based approach since it provides a compact representation of the signal energy and the computation can be performed at low cost [4].

3.3.4 Discrete Cosine Transform (DCT)

In this approach, we analyze the information content of the image in the frequency domain by computing the entropy of the involved DCT coefficients. The DCT provides a compact representation of the signal energy. We apply the 2D DCT on each block. The 2D DCT transforms the input frame to a coefficient matrix where each coefficient represents the degree of which a certain cosine function is present in the input frame. If each block has a high peak only at low frequency and no other significant values, it is not an interesting block. Since the DCT value on location (0,0) in the coefficient matrix is quite different from the other coefficients, we remove this peak to concentrate on the rest of the coefficients. To avoid small values at higher frequencies due to little intensity changes we define a threshold. To this end, we compute the entropy of the DCT coefficients for each block to measure the information content. The entropy E is defined as [28]:

$$E = -\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} P(x,y)\log P(x,y), \qquad (3.4)$$

where N is the size of an image and P is the probability of the intensity value at a certain pixel location (row x and column y). The number of bins in the histogram is specified by the image type. In our case, the input frame is converted to a gray-scale image and we use 256 bins which correspond to the number of gray-scale levels.

For deciding whether each block is a salient region or not, we compare DCT coefficients of the current block with the entropy of this block. If at least one DCT coefficient of each block is above 40% of the entropy of this block, then this block is considered as a salient region. We repeat this procedure for each color component from an RGB signal in each frame. Our DCT-based salient region detection approach is illustrated in Figure 3.3.

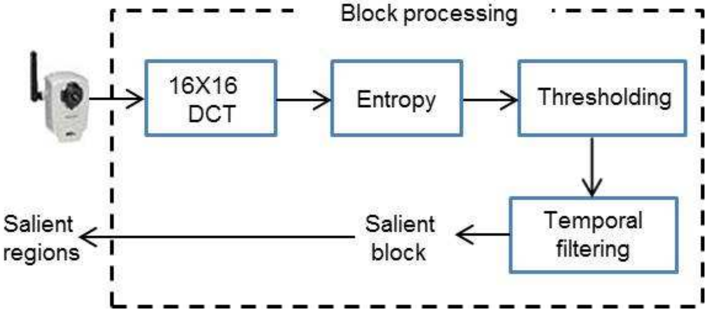


Figure 3.3: Proposed DCT-based salient region detection on the red component from an RGB signal.

3.4 Experimental results

This section starts with describing the datasets: the Caltech Pedestrian Detection Benchmark [89] and our highway video sequence. Then it proceeds with describing the tools and parameters that we have used for our salient region detection approaches in Section 3.4.2. Finally, our results are evaluated qualitatively and quantitatively in Section 3.4.3.

3.4.1 Description of the datasets

For evaluating our salient region detection methods, we apply them on two datasets of videos: the Caltech Pedestrian Detection Benchmark [89] and our highway sequence. The Caltech Pedestrian Detection Benchmark [89] dataset is created for pedestrian detection benchmarking and it is available online [89]. The dataset contains man-made objects, like houses, cars, etc. It consists of approximately 10 hours video taken from a vehicle driving through regular traffic in an urban environment. Frames have the resolution of 640×480 pixels and were computed at 30 frames per second (fps). We have randomly chosen a video fragment containing 1818 frames (about one minute) from the Caltech Pedestrian Detection dataset for visual assessment of the results of the all approaches. Figure 3.4 shows one sample frame of the Caltech Pedestrian Detection Benchmark sequence [89]. We validate our approaches in terms of precision and recall on 30 images from this benchmark sequence. Each frame has to be manually annotated for evaluating the results, that is why we limit the amount of frames for the experiments. We acknowledge that the manual frame annotation may be subjective, as each person will denote a different region of interest. However, it is a promising approach to validate the output of the proposed algorithms. We define empirical thresholds for each technique after extensive experimentation to optimize their performance.

For further analysis of the DCT-based approach, we have also evaluated it on our highway video sequence. This sequence consists of 40 frames with a resolution of 1280×960 pixels and a frame rate of 25 fps. Figure 3.5 illustrates a sample frame of our surveillance highway video sequence.

3.4.2 Settings

The edge-based method is, as mentioned before, based on the 3×3 Sobel edge detection filter. Based on empirical evaluations a threshold for binarizing the sobel detector output is set to 300. As shown in Equation (3.3), the threshold for saliency detection is related to the number of blocks. The number of blocks used for this and connected component is 1000. Based on our experiments this number gives a certain block size which results in enough features to make a decision on saliency.

The Hough-based method transforms each block from the edge map to the Hough space. The edge map is divided into 64 blocks. This is done to ensure that the size of the blocks is large enough to find consistent line seg-

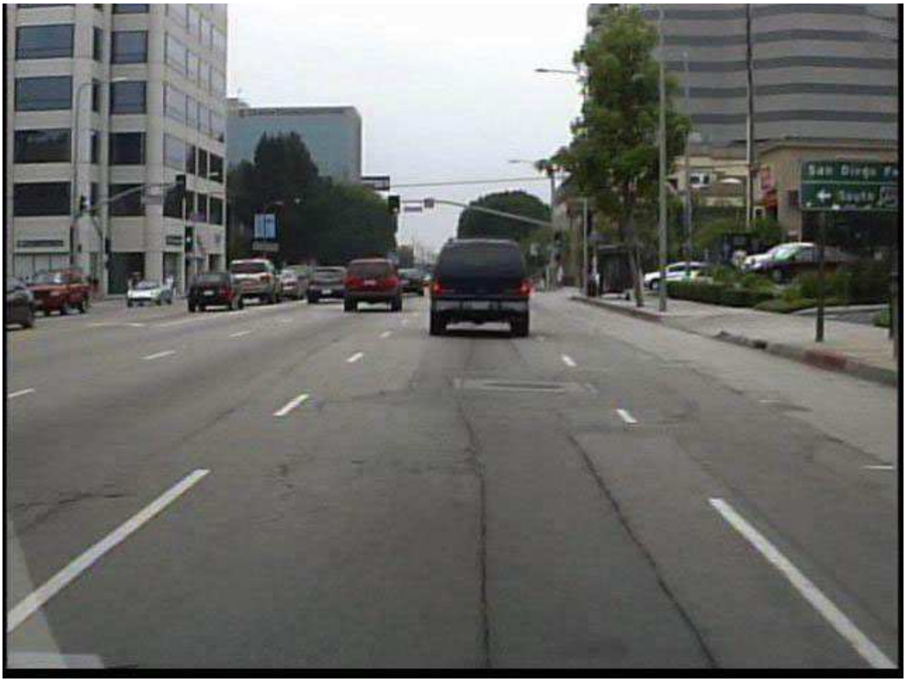


Figure 3.4: Sample frame from the Caltech Pedestrian Detection Benchmark sequence [89].

ments which are also small enough for the output not to be too coarse. The maximum number of lines per block is set to 9. The threshold for the Hough peaks is 30% of the maximum value of the Hough space. Based on empirical evaluations, the number of lines per block should be at least 5 so that the block is chosen as a salient region.

For the DCT method, the 2D DCT is applied to each signal component of the RGB video. Here, each frame is split into 16×16 blocks.

In all the approaches, we use 10-frame sliding intervals for temporal filtering to avoid flickering in the output due to small temporal changes and noise.

3.4.3 Evaluation of the results

All results presented here are obtained with Matlab and its image processing toolbox. Figure 3.6 shows the results of applying the proposed salient region

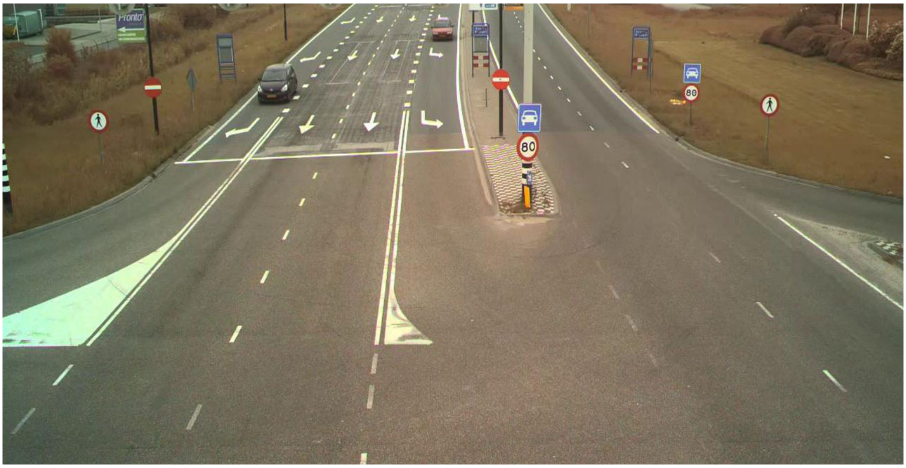


Figure 3.5: Frame from our surveillance highway video sequence.

detection techniques on a sample frame of the Caltech Pedestrian Detection Benchmark sequence [89]. It should be noted that due to the fact that the number of blocks of the Hough-based method is smaller than other methods, the resulting image is more coarse for this method. As shown in Figure 3.6, compared to the ground truth in Figure 3.6 (b), all methods perform similarly well. However, it appears that the false negatives produced by the Hough- and DCT-based methods are the lowest. Figure 3.7 illustrates the results of applying the proposed salient region detection techniques on another sample frame from the Caltech Pedestrian Detection Benchmark sequence [89]. This frame contains a different scenery so that we can better assess the quality of the results.

Table 3.1 presents the average precision and recall rates of the salient region detection approaches on 30 images of Caltech Pedestrian Detection Benchmark [89]. It is indicated that the edge and connected component-based methods have the lowest recall of 50% and 57%, respectively. It can be also observed that that the results of the salient region detection obtained from the DCT-based approach with precision of 61% and recall of 79%, are similar to the results of the Hough-based technique (with precision of 61% and recall of 80%). However, as it will be shown in Chapter 5, the Hough-based method is the slowest algorithm in executation. Therefore, we conclude that the DCT-based approach performs better compared to the other approaches. Consequently, for further analysis of the DCT-based approach, we also evaluate it on our surveillance highway video sequence. For this video sequence we obtain precision and recall of 39% and 75%, respectively. The average precision and recall rates of the DCT-based approach for 70 images of both the Caltech Pedestrian Detection Benchmark [89] and our dataset are 50% and 77%, respectively. For our surveillance application, we consider the recall rate to be more important than precision because we aim at extracting as many salient regions as possible from the image. For surveillance applications, it is important to present to the security officer all salient regions that are present in a given frame, so that no important information is missed.

Approach	Recall (%)	Precision (%)
Edge-based	50	62
Connected component-based	57	60
Hough-based	80	61
DCT-based	79	61

Table 3.1: Recall and precision rates for the salient region detection algorithms on 30 frames from the Caltech Pedestrian Detection Benchmark [89].

In order to benchmark our approach, we compare our DCT-based salient region detection approach with Rahtu *et al.* [90]. The proposed saliency measure approach by Rahtu *et al.* [90] is formulated using a statistical framework and local feature contrast in illumination, color, and motion information. The resulting saliency map is then used in a Random Forest model to define a segmentation approach that is based on an energy minimization technique aiming at recovering salient objects. The method is efficiently implemented by using the integral histogram approach and graph cut techniques. The method of Rahtu *et al.* is applied here on still images. It should be noted that the Rahtu approach aims at segmenting a single salient object per frame. The aiming at a single object per frame is common in the saliency detection area, where we assume that a frame can contain a number of salient objects. Therefore, to make Rahtu's approach suitable for our application, we applied this approach per block and the results reported here are according to their best outcomes.

Figure 3.8 shows the results of applying our DCT-based approach and the approach proposed by Rahtu *et al.* [90] on a sample frame of our surveillance highway video sequence. Table 3.2 presents the results of applying the DCT-based approach, compared to Rahtu's algorithm on 70 images of our dataset. It can be observe that by using the DCT-based approach, the recall is increased from 25% to 36% and the precision is increased from 63% to 85%, as compared to Rahtu's approach. Evidently, our DCT-based approach outperforms the algorithm of Rahtu *et al.*

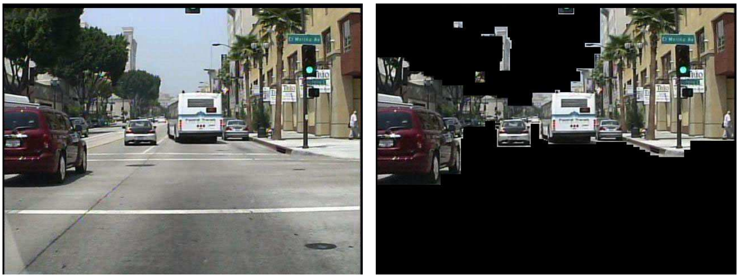
Approach	Recall (%)	Precision (%)
Rahtu *et al.* [90]	63	25
Our DCT-based	85	36

Table 3.2: Recall and Precision rates for the salient region detection methods.

3.5 Conclusions and discussion

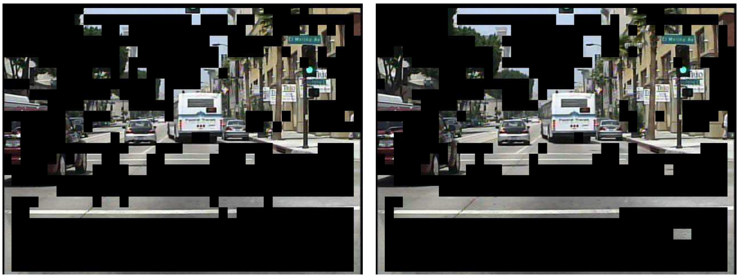
In this chapter, we have presented our research on detecting salient regions for outdoor surveillance video. We have introduced four salient region detection techniques which are based on sum of edge pixels, number of connected components, the number of straight lines found by the Hough transform and the entropy of DCT coefficients. It has been shown that the Hough- and DCT-based methods are better than edge and connected component-based methods in terms of recall and precision. It has further been indicated that the results of the salient region detection obtained from the DCT-based approach, are similar to the results of the Hough-based technique. However, it will be shown in Chapter 5 that, the Hough-based method is the slowest algorithm. Therefore, we have concluded that the DCT-based approach performs better compared to the other approaches. The DCT concentrates the signal energy to lower frequencies, so that explicit high-frequency energy in a block indicates saliency in that block. Furthermore, with a better image quality we may improve the performance of the approach besides considering other features such as motion. In order to benchmark our DCT-based approach, we have compared it with the algorithm of Rahtu *et al.* [90]. The experimental results have shown that our DCT-based approach outperforms the approach of Rahtu *et al.* with approximately 44% in precision and 35% in recall. We can explain this difference in performance by the fact that our algorithm design aims at multiple salient regions per frame, instead of a single region, so that our approaches can directly be applied to surveillance application.

The computational complexity of the DCT-based approach will be addressed in Chapter 5. Furthermore, to measure the potential efficiency gain, we cascade our DCT-based salient region detection with a typical object detector [91] as used in surveillance cases. Our hypothesis is that detection of salient regions provides a significant gain in efficiency in terms of the number of pixels needed to be explored, compared to analyzing the complete scene. The evaluation of our hypothesis will be addressed in Chapter 6. However, prior to discussing complexity and the related use cases of our DCT-based saliency detector, in the next chapter another aspect of region analysis, i.e. semantic region labeling approach, is presented.



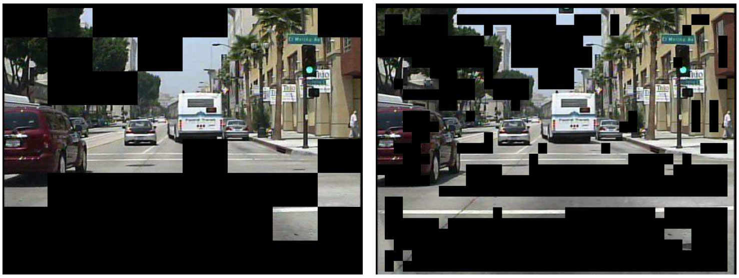
(a) Caltech Pedestrian Detection Benchmark sequence [89]

(b) Ground truth for this frame



(c) Edge-based method

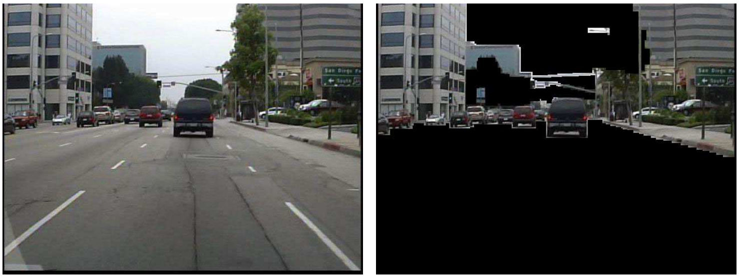
(d) Connected components-based method



(e) Hough-based method

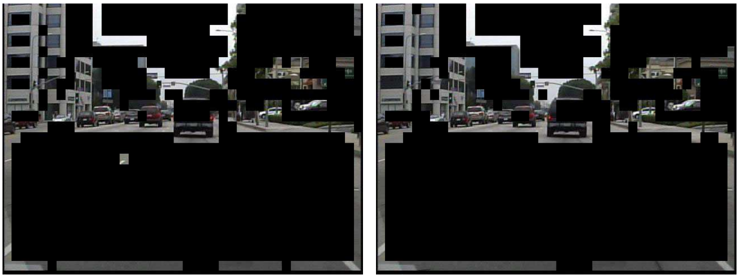
(f) DCT-based method

Figure 3.6: Results obtained with the proposed salient region detection techniques on a sample frame, compared to other methods.



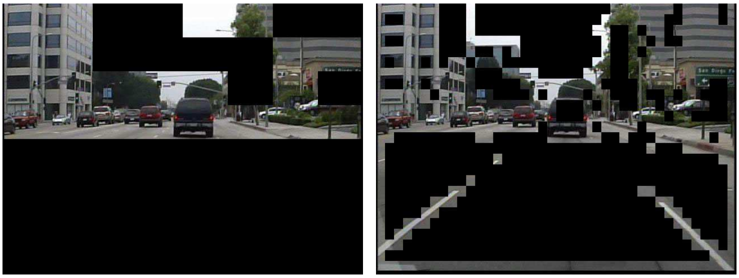
(a) Caltech Pedestrian Detection Benchmark sequence [89]

(b) Ground truth for this frame



(c) Edge-based method

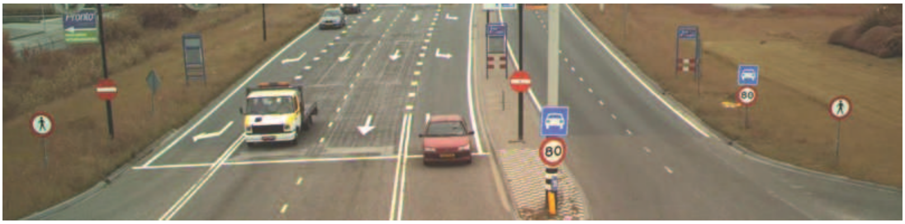
(d) Connected components-based method



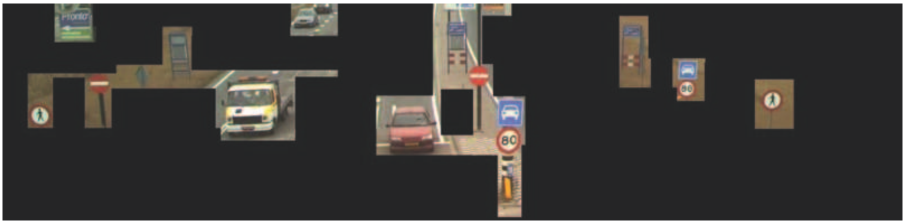
(e) Hough-based method

(f) DCT-based method

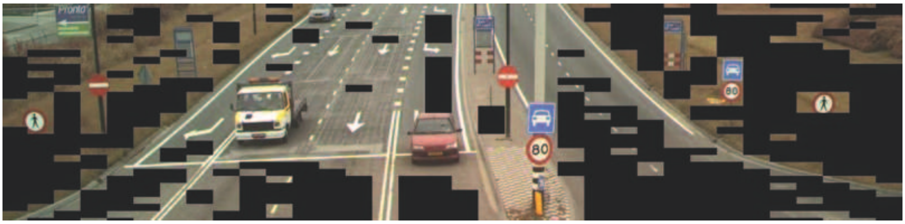
Figure 3.7: Results obtained with the proposed salient region detection techniques on a sample frame, compared to other methods.



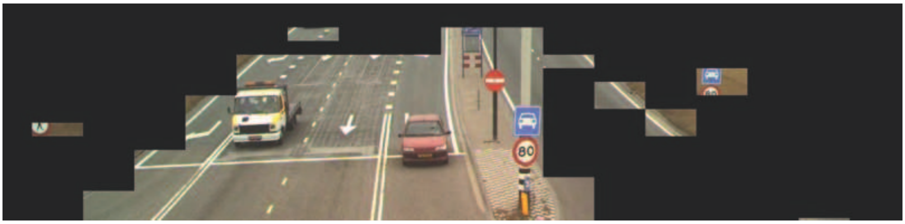
(a) Our highway video sequence



(b) Ground truth for this frame



(c) Our DCT-based salient region detection approach



(d) Approach of Rahtu *et al*. [90]

Figure 3.8: Results obtained with different salient region detection techniques on a sample frame on our highway video sequence.

4

Semantic region labeling

4.1 Introduction

Chapter 3, has addressed the problem of salient region detection. We have introduced and compared the four salient region detection techniques, which are based on the Sobel edge detector, connected components, Hough transform and the DCT. After salient region detection, the next research question is the semantic labeling of the region, which is essential for automated surveillance scene analysis.

This chapter is dedicated to the region level of the scene understanding, i.e. semantic region labeling. Semantically labeled regions are helpful for two general applications: (1) to detect an arbitrary region in an image for general purposes, or (2) to improve the detection of an object-of-interest in a scene. The latter type of detection improvement is because the location of the object can be linked to a semantic region that is likely to contain that specific object. For example, in port surveillance, the detection of the water region can help to detect ships, as ships can only travel within the water region.

This chapter, considers the semantic region labeling approach in two important ways, i.e. (1) labeling of each individual specific region, e.g., separate approaches to label sky, water and road regions, and (2) trying to develop developing a general framework for performing automatic semantic labeling task, e.g. labeling an input scene with multiple semantic labels such as sky, road and vegetation at the same time.

This chapter starts with Section 4.2 which reviews the related work for the first case for specific semantic labeling of a region. Section 4.3 presents detection algorithms for three specific regions which are most often present in an outdoor scene, i.e. sky, water and road regions. For sky detection, in Section 4.3.1 we adopt a detection algorithm from Zafarifar *et al.* [92], based on a probability map that jointly uses a color model, texture properties at multi-resolution scale and the vertical position. For water detection, in Section 4.3.2 we consider a detection algorithm from Liu *et al.* [93] where color is analyzed, and SVM is used to classify the regions, using the RGB pixel values. For road detection, in Section 4.3.3 we introduce a motion-based approach to annotate roads and to restrict the computationally heavy search for moving objects to the areas where the motion is detected. Section 4.4 presents our results for the specific semantic region labeling approaches. Conclusions and discussion related to the specific semantic region labeling approaches are provided in Section 4.5. Section 4.6 continues this chapter by focusing on the second case, where we introduce the details of our generic semantic labeling approach which is based on combining the local features and spatial contextual cues. Experimental results related to the generic semantic region labeling approaches are provided in Section 4.7. Conclusions and discussion related to the second case are provided in Section 4.8.

4.2 Overview of semantic labeling approaches for specific regions

This section reviews the related work for detecting three specific regions that frequently occur in an outdoor scene, i.e. sky, water and road. The challenge here is to determine which features are needed to distinguish between the specific regions and their surrounding areas.

Sky detection provides the context information for further image analysis, while it helps to extract information about e.g. weather and illumination conditions. Due to sky variations, systems that restrict themselves to blue sky detection, have limited practical value [94]. Previous work on sky detection includes a system [95], based on calculating an initial ”sky belief map” using color values and a neural network, followed by connected-area extraction. Such a system is not suitable for the requirements of video surveillance applications concerning spatial consistency. For example, due to connected-area extraction, patches of sky may be rejected when their size is reduced during a camera zoom-out. A second system proposed in [96] is based on the assumption that sky regions are smooth and are normally found at the top of the image. Using predefined settings, an initial sky probability is calculated based on color, texture and vertical position, after which the settings are adapted to regions with higher initial sky probability. This system is suitable for video applications. However, due to its simple color modeling, this method often leads to false detections, such as classifying non-sky blue objects as sky, and false rejections, like a partial rejection of sky regions when they cover a large range in the color space.

Water detection is another research direction for outdoor scene analysis. This analysis involves several influencing factors, like day/night time, reflection at the water surface, relative size of the water region, wave state and possible occurrence of objects at the water surface [94]. Matthies *et al.* [97] developed a color image classifier based on a mixture of Gaussians, to exploit means and standard deviations of brightness and saturation. The authors trained this classifier on water regions in the RGB color space. In [98], active learning and mean-shift based image segmentation were combined to classify water regions. Rankin *et al.* [99] used color image classification to recognize water by its reflection of the sky. However, when still water reflects other objects, such as trees, hills or buildings, the performance of the algorithm may deteriorate.

Another frequently occurring region in outdoor video scenes is the road region. The detection of road regions is essential for increasing the safety of pedestrians within overall traffic by e.g. abnormal behavior detection and traffic analysis. In the past decades, several approaches for road detection have been proposed. According to [100] road detection algorithms can be categorized into three main classes: model-based, region-based, and feature-based techniques. Model-based techniques are quite robust. However, most model-based techniques are established on some critical and strict geometrical assumptions. An example of the model-based approach is based on a Markov framework to detect road lanes in video sequence [101]. This approach allows to find a traffic lane of a road, but not the whole road, which may be composed of multiple lanes. Region-based techniques use learning algorithms. In this case, it is important and yet challenging to define a small set of good features to be extracted from images and to compose an optimal training set. Generally speaking, a feature-based approach is more precise than other techniques. However, this requires that the road has well-painted markings in order to be detected [100]. Usually, two particular features, namely color and texture, are used to extract the road region. The color and texture in various roads are normally not different. It should be noted that due to the uncontrolled illumination conditions, the color of a road varies with time [102]. Mezaris *et al.* [103] proposed an algorithm for road detection that is based on exploiting the color and motion information of the MPEG-2 macroblocks. Currently, combining spatial and temporal features (i.e. motion) is one of the most attractive new technologies [102].

The next section presents our detection algorithm choices for recognizing sky, water and road regions.

4.3 Semantic labeling of specific regions

This section starts with describing our sky detection algorithm. Then, Section 4.3.2 presents our water detection approach and Section 4.3.3 introduces our road detection technique.

4.3.1 Sky detection

For sky detection we have adopted an algorithm from Zafarifar *et al.* [92]. We evaluate the results of this algorithm on two different datasets, i.e. the dataset of Schmitt [104] and the WaterVisie dataset. We also compare its results with a similar approach proposed by Schmitt [104]. Let us briefly address the sky detection approach proposed by Zafarifar.

This sky detection algorithm assumes that sky regions are smooth and are located at the top of the image. An initial sky probability map is calculated based on color, texture and the vertical position of pixels, after which the features from high-probability areas are used to compute a final sky probability. This algorithm considerably improves false detection/rejection rates. The improvements are primarily due to an extensive multi-scale texture analysis, adaptive thresholds and a spatially-adaptive color model. Next to position and color features, the key assumption of the system is that sky has a smooth texture and shows limited luminance and chrominance gradients, which are different in the horizontal and vertical directions. Using predefined settings for the color, vertical position, texture, and horizontal and vertical gradients, an initial sky probability is calculated for each image pixel [105] by the following expression:

$$P_{sky} = P_{\text{color}} \times P_{\text{position}} \times P_{\text{texture}} \times Q_{\text{sky}}, \tag{4.1}$$

where the color probability P_{color} is computed by a three-dimensional Gaussian function in the YUV color space, having a fixed variance and is centered at a predetermined color. The parameter for the position probability P_{position} is defined by a function that emphasizes the upper parts of the image. For the texture probability P_{texture}, a multi-scale analysis is performed on the image, using an analysis window of 5×5 pixels. The initial sky probability map needs to be segmented by a threshold in order to create a map of regions with high sky probability. Simple measures for threshold determination, such as using the maximum of the sky-probability map can cause false detection. For example, small objects at the top of the image with color and texture similar to sky regions (i.e. with high sky probablity value) can be detected as sky regions. In order to avoid this problem, Zafarifar *et al.* [92] propose a robust method that takes both the size and the probability of sky regions into account, by computing an adaptive threshold and a global sky confidence-metric Q_{sky}.

4.3.2 Water detection

For water detection we have adopted an approach from Liu *et al.* [93] which consists of three stages: (1) segmentation, (2) visual feature extraction and (3) classification. We now briefly discuss these stages of the water detection algorithm.

Stage 1 of water detection: segmentation. In this stage, each image is segmented in terms of the color uniformity of a region by applying a graph-based segmentation method. The usage of the normalized color space and also an additional parameter measuring the intensity difference of two neighboring pixels are introduced to a graph-based segmentation, in order to adapt the algorithm to our application. The graph-based segmentation method was chosen because it was proven by Liu *et al.* [93] to be suited for real-time processing and has sufficient accuracy. By carrying out the segmentation stage, we aim at increasing the robustness of detection and decreasing the calculation complexity in region recognition. The details of this stage will be presented in Section 4.6.1, where we introduce our generic region labeling.

Stage 2 of water detection: visual feature extraction. In this stage, visual features of each region, i.e. the intensity values (RGB), are extracted to be used in the third stage.

Stage 3 of water detection: classification. In the third processing stage of water detection, a Support Vector Machine (SVM) classifier is applied to detect water regions. The classifier is trained off-line, based on the image samples captured at a harbor in different weather conditions.

We have evaluated this algorithm on the WaterVisie dataset which is presented in Section 4.4. It should be noted that the basic components of this approach are used in our generic semantic region labeling approach which is discussed in more detail in Section 4.6.

4.3.3 Road detection

In our road detection approach, we aim at developing a novel real-time approach to detect roads in video sequences, using a combination of two main features of a road: the motion of objects on the road and the straight lines indicating the boarders of the road. Our algorithm contains three stages: (1) heat-map-based motion analysis, (2) straight-line detection and (3) combination of features.

Stage 1 of road detection: motion analysis. To analyze and visualize motion in a scene, we apply the concept of a heat-map. A heat-map is a 2D histogram indicating main regions of motion activity [106]. The motion heat-map represents "hot" and "cold" areas on the basis of motion intensities. The hot areas are the zones of the scene where the motion is large. The cold areas are regions of the scene where the motion is small. This map can be designed from the accumulation of binary blobs of moving objects, which are extracted following the background subtraction method [106]. We subtract the background by thresholding absolute differences between frames. The obtained heat-map can be used as a mask to define regions of interest. The use of the heat-map image improves the quality of the results and reduces processing time, which is an important factor for real-time applications.

Besides the motion analysis based on generating the heat-map, we also identify the direction of the movement in the scene using the optical flow [106] approach. By applying the optical flow technique, we track a moving object over succeeding frames. For this tracking, we employ the Kanade-Lucas-Tomasi (KLT) feature tracker [106]. After matching an object between frames, the result is a set of vectors that indicate the direction of the moving object.

Stage 2 of road detection: straight-line detection. Real-world roads contain many segmented lines or boundary lines. The Hough transform is known as an algorithm that can extract lines efficiently [4]. We have used a standard Hough transform and the details of this method were discussed in Chapter 2.

Stage 3 of road detection: combination of features. In this stage of the road detection approach, we combine motion and straight-line features. To do this, we consider all pairwise combinations of the lines found by the straight-line detector. For each pair of lines, we sum the heat-map values of the pixels that lie between these lines. If the obtained value is above a preset threshold, it means that the selected pixels are inside the "hot" area. We classify all pixels between those two lines as part of a road. The steps of our road detection approach are summarized in Algorithm 1.

4.4 Experimental results for semantic labeling of specific regions

To measure the performance of our approaches, we use the Coverability Rate (CR), which indicates how much of the specific region, i.e. sky, water or road, is detected by the algorithm [104]. This rate is computed by Equation (4.2)

Algorithm 1 Road detection

```
for The whole video sequence do
    Measure pixel-based motion related to previous frame to extract binary blobs
    Accumulate binary blobs of moving objects to define "hot" and "cold" areas
end
for each frame do
    Apply Sobel operator
    Apply Hough transformation
    Sum the heat-map values between each pair of lines
    if The heat-map values exceed the threshold then
        classify the pixels between the lines as a road region.
    end
end
```

as:

$$CR(O, GT) = \frac{|O \cap GT|}{|GT|}, \tag{4.2}$$

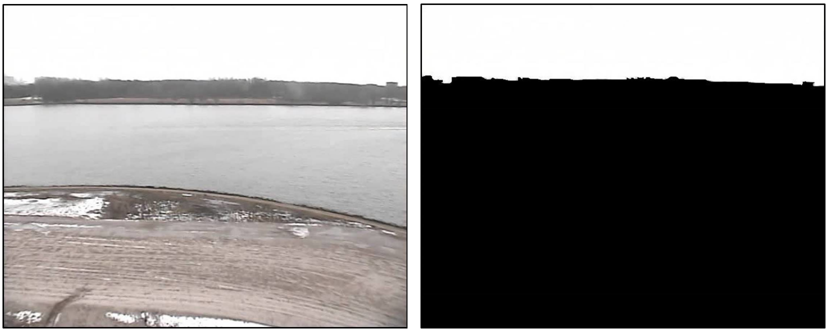
where Ground Truth (GT) is manually annotated water, sky or road region, and O is the area detected as sky, water or road. It should be noted that other metrics such as error rate have also been used in the literature [104]. However, for our application, we consider the accuracy or CR (recall) to be more important than error rate (precision), since we aim at extracting ROI regions for further analysis of the regions. For surveillance applications, it is important to provide all ROI regions that are present in a given frame so that no important information is missed.

Let us first discuss the sky detection results. In order to analyze the performance of the sky detection algorithm, we have compared its results with two datasets: the WaterVisie dataset and the dataset of Schmitt [104]. The WaterVisie dataset presents mostly outdoor surveillance scenes that contain vegetation, water and ships. The dataset of Schmitt is composed of urban scenes where we observe buildings, pavements and little vegetation. We briefly summarize an alternative approach from literature introduced by Schmitt [104] for the sake of performance comparison. The algorithm is based on the analysis of color, position and shape properties of homogeneously colored, spatially connected regions. In the first steps of the algorithm, the input image is smoothed and segmented by a region growing segmentation technique. Each segment is analyzed individually regarding its average color, and a sky probability is attached. In a final step, spatial information typical for urban scenes is added to the probability map and all segments are classified into sky and non-sky.

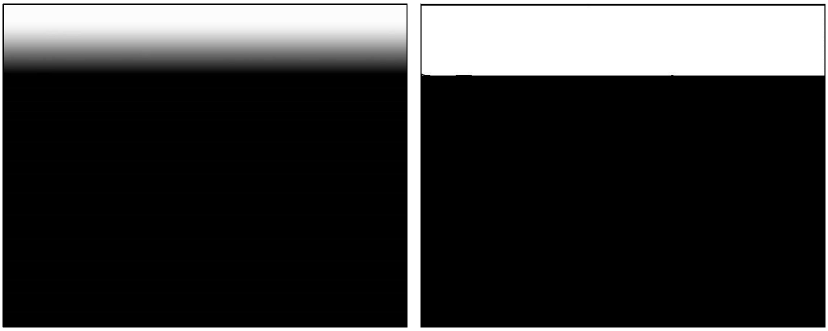
For quality evaluation of our sky detection, we manually annotate sky on 17 images of the WaterVisie dataset and 179 images of the dataset of Schmitt

[104]. Then, the images of Schmitt's dataset are divided into 100 images for training and 79 images for testing. All WaterVisie images are only used for testing. Afterwards, our sky detection algorithm [105] is trained and tested. Figure 4.1(a) shows the original image of the WaterVisie dataset, which is one frame extracted from a video sequence. As can be observed, this dataset does not provide sufficient information in terms of color. Figure 4.1(c) visualizes the corresponding probability map of the sky region of that image, which is obtained by applying our sky detection algorithm to the original image, and Figure 4.1(d) shows the result of applying a threshold to the probability map.

Figure 4.2(a) shows an original image of the dataset of Schmitt. It can be



(a) Original image (b) Manually annotated sky region

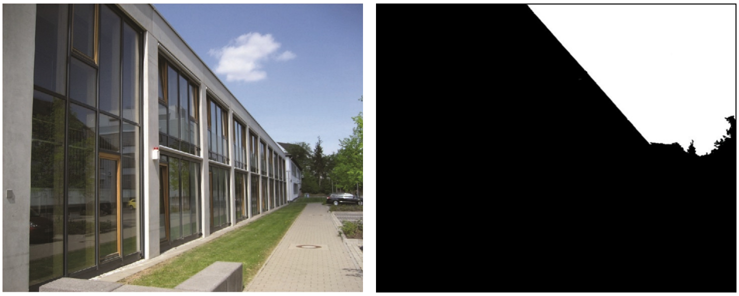


(c) Probability of the sky region (d) Result of the sky detection after thresholding

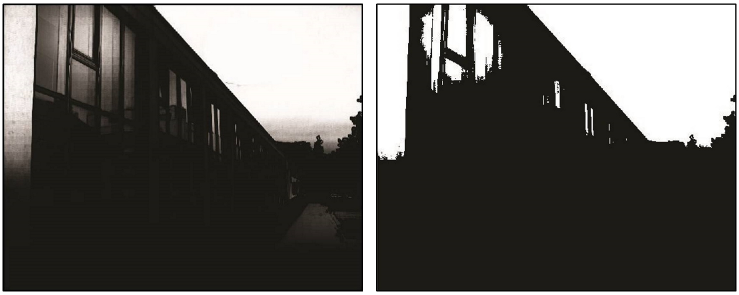
Figure 4.1: Example of sky detection algorithm on the WaterVisie dataset (white color indicates sky).

seen that this image does not provide sufficient information in terms of blue color, due to clouds. Figure 4.2(c) visualizes the probability map of the sky region which is achieved by our sky detection algorithm on the original image, and Figure 4.2(d) shows the result of applying a threshold to the computed probabilities. The CR for this image is 99%. Table 4.1 shows our sky detection results for the WaterVisie dataset and the Schmitt's dataset. The performance of our algorithm is comparable to Schmitt algorithm and sometimes slightly better. Schmitt *et al.* [104] have reported that in 80% of all images lead to a CR that is above 90%.

Regarding water detection, we evaluate our approach on the WaterVisie dataset. We manually annotate water on 40 images of the WaterVisie dataset and we use 20 images for training the classifier and 20 images for testing purpose. Figure 4.3 shows the original image and the obtained detection results.



(a) Original image, (b) Manually annotated sky region

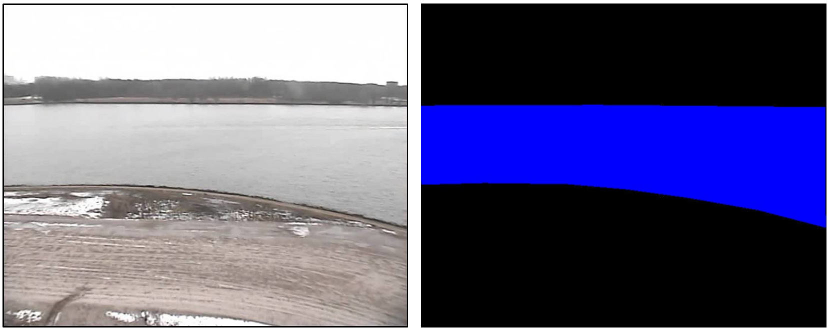


(c) Probability map of the sky region (d) Thresholded map of the sky region

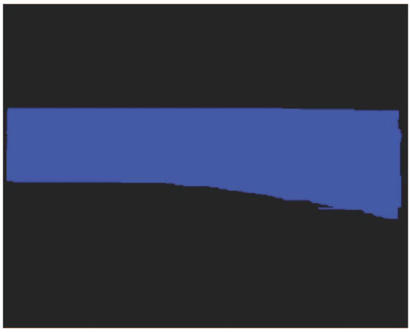
Figure 4.2: Example of Sky detection result on the Schmitt dataset [104] (white color indicates sky).

Figure 4.4 shows an alternative case, where a ship is entering the scene. In both cases the water region is correctly found and the obtained average of the CR for water detection in all annotated images is about 96.6% (see Table 4.1).

Regarding our road detection approach, the video sequence used for experiments contains a straight road with two lanes and some vegetation at each side of the road with a resolution of 720 × 576 pixels. We can observe two neighboring roads that appear in the upper corners of the scene (Figure 4.5(a)). Figure 4.5(b) shows the corresponding heat-map obtained for this video sequence. This heat-map was obtained using 100 consecutive frames of the sequence. We expect to obtain a similar heat-map while using much

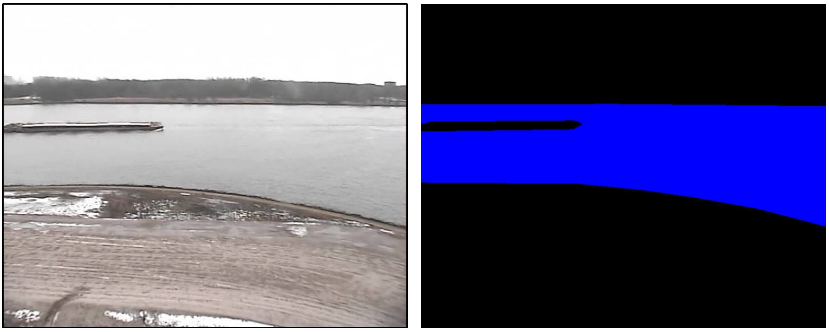


(a) Original image (b) Manually annotated water region

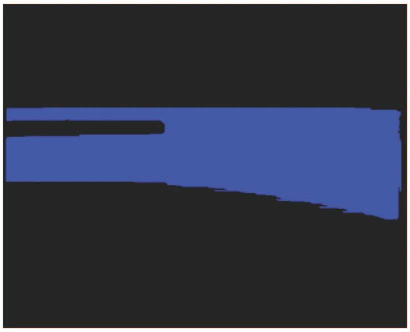


(c) Result of water detection algorithm

Figure 4.3: Example of the proposed Water detection algorithm on the WaterVisie dataset (blue colors indicate the water region).



(a) Original WaterVisie image with ship (b) Manually annotated water region

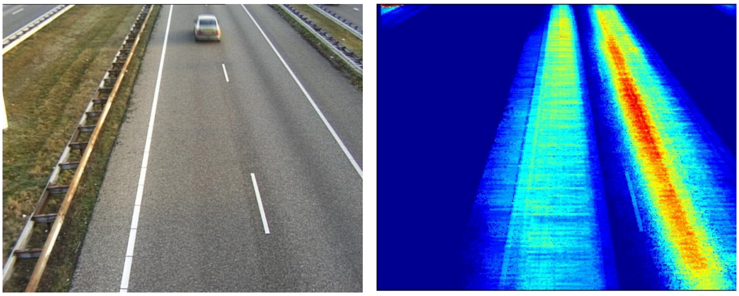


(c) Result of water detection algorithm

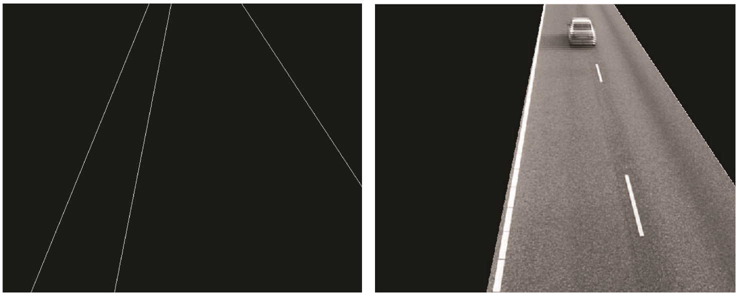
Figure 4.4: Example of the proposed Water detection algorithm on the WaterVisie dataset (blue colors indicate the water region).

less frames. In our experiment, we consider first 3 peaks of the accumulator array with Hough transformation results, to calculate three lines which are the optimal choice for our scene. Figure 4.5(c) presents the result of applying a Hough transform to the selected frame, where the result contains an undesired straight line. To remove this line, we consider the heat-map with "hot" and "cold" areas, which illustrates the value of activity in each pixel. We apply a background subtraction technique on our video sequence to compute a heat-map. The threshold that we impose on the gray-scale values during background subtraction is 120. Figure 4.5(d) shows the final result of the proposed method when the heat-map and the detected straight lines are combined by accumulating the motion values in between the straight lines in the whole potential road area. If the accumulated value is above an empirical threshold, that area between the straight lines is classified as road. The results of the road detection algorithm are promising with 97% in this single highway video sequence (see Table 4.1).

Figure 4.6(b) shows the resulting heat-map on the WaterVisie dataset.



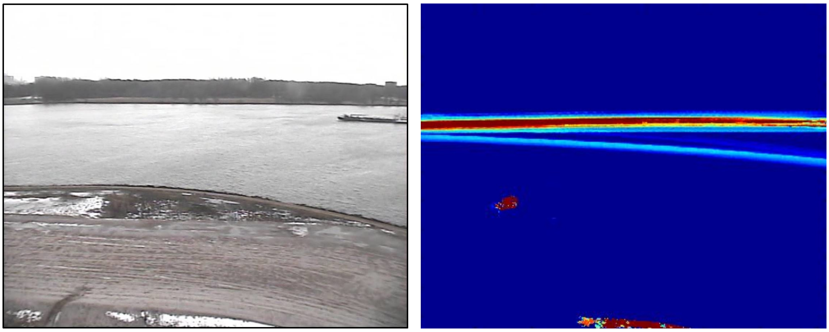
(a) Original image (b) Heat-map of the road



(c) Hough transform (d) The obtained road detection

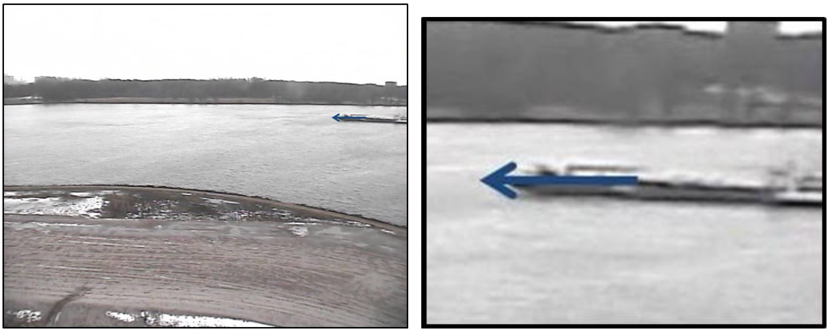
Figure 4.5: Exmple of Hough-based road detection results on our highway video sequence.

There is a circular red area in the middle of the image which is caused by water movement and another red area at the bottom of the scene where the motion is mainly caused by vegetation movements in these areas. We apply the KLT technique to pairs of smoothed neighboring frames of our dataset, after which velocity vectors of each pixel are computed. If velocity vectors are zero, we ignore them to achieve better visualization. The result of the motion detection is shown in Figure 4.6(c). Figure 4.6(d) shows the direction of the moving object.



(a) Frame with a ship

(b) Heat-map on video of a moving ship (the red color indicates regions with the most movement)



(c) Optical flow KLT on a moving ship, arrow shows the direction of ship's movement

(d) Magnified arrow for the direction of the ship's movement

Figure 4.6: Example of using optical flow KLT on a video including a moving ship.

4.5 Discussion and conclusions for semantic labeling of specific regions

We have presented our research on detecting three specific regions which occur most frequently in an outdoor scene, i.e. sky, water and road.

For sky detection, we have adopted a model-based detection algorithm from earlier work, based on a probability map that jointly uses a color model,

Approach	Dataset	Coverability Rates (%)
Sky detection	WaterVisie	98
Sky detection	Schmitt [104]	> 90
Water detection	WaterVisie	96.6
Road detection	highway sequence	97

Table 4.1: Average of the obtained Coverability Rates (%) for three specific region labeling approaches on different datasets.

texture properties at a multi-resolution scale and the vertical position of potential sky pixels [105]. To evaluate the results, the average CR values have been calculated for two different datasets: the WaterVisie dataset and the dataset of Schmitt [104]. It has been shown that the algorithm of Zafarifar [105] performs well for both datasets, showing a CR of 98%. It has further been indicated that this algorithm performs equally well as the algorithm of Schmitt, while being more flexible, because it uses adaptive thresholds and does not limit itself to a particular type of scene. We have concluded that the adaptive model-based technique [105] is more reliable for detection of various types of sky.

Regarding water detection, we have also evaluated the dedicated algorithm in which segmentation and region recognition are combined. The obtained CR average for water detection is about 97% for the WaterVisie dataset. The basic components of this approach are used in our generic semantic region labeling algorithm which will be discussed in more detail in the next section.

Regarding road detection, we have designed a novel approach by combining two reliable features which are based on motion and straight-line detection. It has been indicated that our results obtained with this algorithm yield a CR of 97%.

Since we have exploited various features in each detection algorithm to classify sky, water and road regions, we conclude that developing specific algorithms for each specific region is not adaptive to new semantic region types and we need to re-design a new approach for each new region type. Therefore, our research study for semantic region labeling has been led into developing a general framework for performing automatic semantic labeling task. To this end, we use common features for several natural outdoor regions to design the framework.

The chapter continues by introducing the details of the generic semantic labeling approach, which is based on combining both the local features and spatial contextual cues. Section 4.7 presents the experimental results of the generic labeling approach on three datasets: our own dataset, LabelMe [107] and WaterVisie [108]. The general approach which contains 5 classes (sky, vegetation, road, water, construction) plus the class "unknown". Our method is also compared quantitatively and qualitatively with two state-of-the-art approaches [109][110]. Conclusions and discussion related to the generic semantic region labeling are provided in Section 4.8.

4.6 Generic semantic region labeling

The related research for semantic region labeling approaches as well as their processing stages have been reviewed in detail in Chapter 2. We note here that one of the challenging aspects of automatically labeling image regions is taking the contextual information into account which is not often considered in prior research. The context provides a rich source of information that can help to improve a scene analysis performance and reduce ambiguities of very local scene information [1]. Although local features like color and texture for regions such as sky and water are instrumental for understanding, they are typically not uniquely determining the semantic meaning of these regions. For a reliable semantic labeling of regions, the labeling task should account for the contextual information at both local and global scene levels. Therefore, we propose a reliable generic region labeling approach where information at both global and local scene levels is incorporated for. Our generic semantic region labeling algorithm involves three stages, as depicted in Figures 4.7 and 4.8, which are discussed here.

Stage 1 of generic semantic region labeling: uniform regions. In this stage, the image is divided into several regions with uniform color using the graph-based segmentation method.

Stage 2 of generic semantic region labeling: feature extraction. A global contextual information as well as pixel-based features (HSV color space and a group of Log-Gabor features) of each segmented region are extracted. Regarding global feature extraction, two methods are proposed based on: (1) Spatial Context (SC) in which the normalized vertical position for each pixel is calculated; (2) Global Region Statistics (GRS) in which intervals for mean and standard deviation of vertical positions of each specific region are obtained.

Stage 3 of generic semantic region labeling: classification. For this stage, our algorithm employs two concepts in a sequential order.

- *Multiple-SVM (one vs. all).* For each region class, an off-line separately trained SVM is used to classify that region. Given the feature analysis of

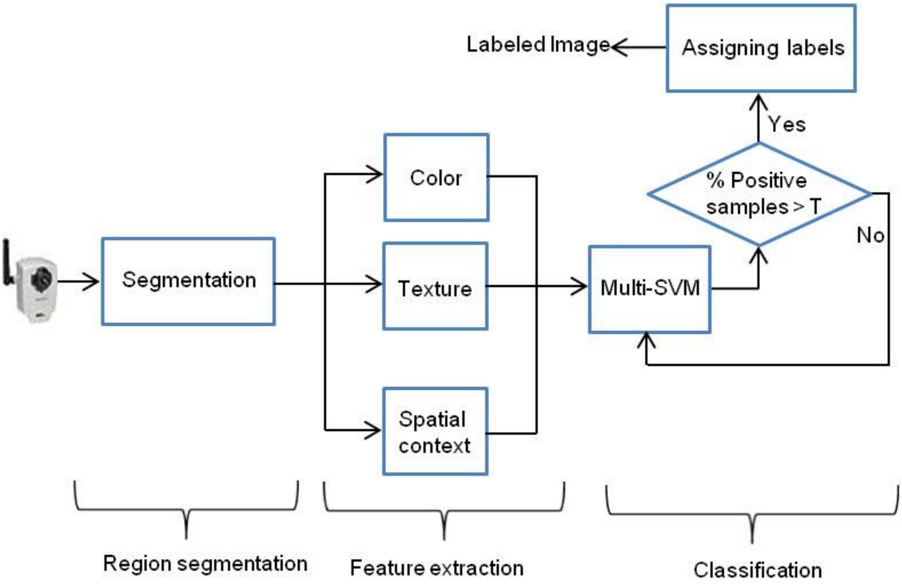


Figure 4.7: Proposed gravity-based region labeling approach.

the previous stage, color, texture and SC (normalized vertical position) are used for learning each region class separately. We call the extensive use of vertical information a *gravity-based model*. However, this is not sufficient for region classification. Therefore, the stage of *Assigning labels* is also included.

- *Assigning labels*. For each particular class, we measure the percentage of pixels classified as belonging to that class in a given region. We assign a specific label to a region when the percentage of positively classified pixels in this region is above a threshold.

Figures 4.7 and 4.8 depict the two instantiations of our generic region labeling approach. In Figure 4.7 the first method is depicted in which SC is added in the form of the gravity-based model to the feature extraction stage. Figure 4.8 depicts the second method in which GRS is used in the classification stage.

In the gravity-based model an SVM is trained on HSV color space, a group of Log-Gabor features and the SC information. However, in the GRS-based method, only the HSV color space and a group of Log-Gabor features are used for training the SVM and SC information is avoided for the training. Subse-

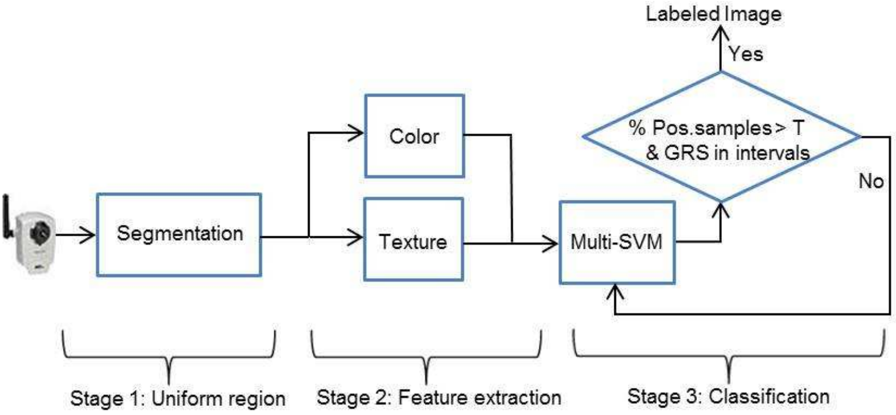


Figure 4.8: Our GRS-based region labeling approach.

quently, the intervals for mean and standard deviation of vertical positions are used in the *Stage 3* for classification. This is a way to include position information at a region level without using the local gravity-based model. Section 4.6.3 - B describes the GRS-based region labeling approach (Figure 4.8) in more detail.

4.6.1 Stage 1 of generic semantic region labeling: uniform regions

We adopt an efficient graph-based segmentation from [111] as pre-processing in our region labeling to achieve two objectives: (a) oversegmenting an image so that we can group the segments into semantically meaningful regions, and (b) performing fast segmentation to support a real-time application in surveillance systems [108]. The basic idea of the graph-based method is that pixels within one region are closer in color space than pixels from different regions. We now define the segmentation stage more formally [111]. The image is first blurred using a Gaussian filter with the standard deviation σ. In this method, for each pair of neighboring pixels i and j, there is an edge with an Euclidean weight $w_{i,j}$ which is specified by:

$$w_{i,j} = \sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2}, \tag{4.3}$$

where R_i, G_i, B_i are the RGB color values of the pixel i.

For the proper operation of the algorithm, two weights are defined. First, intra-region weight (W_A) of region A is defined as the maximum edge weight within the region. Initially, each pixel is regarded as a region, and the initial threshold τ for each region is set to $\tau_A = \kappa/|A|$, where $|A|$ is the number of pixels in the region A. Note that $|A| = 1$ if region A contains one pixel and κ is a constant parameter controlling the merging, such that a larger value results in larger segments. The second weight is inter-region weight $W_{m-inter}(A, B)$ of a pair of regions A and B, which is defined as the minimum of intra-region weights of the involved regions A and B. This weight now becomes:

$$W_{\text{m-inter}}(A, B) = \min(W_A + \tau_A, W_B + \tau_B). \tag{4.4}$$

Regions A and B are merged into a new region if they satisfy the following condition:

$$W_{\text{m-inter}}(A, B) < W_{\text{m-intra}}(A, B). \tag{4.5}$$

If the merging takes place, it is evident that the weight for the merged region $(A \cup B)$ becomes then identical to $W_{(A \cup B)} = W_{m-inter}(A, B) + \tau_{(A \cup B)}$. Note that very small regions are merged based on the minimum region size (S_{min}), even if the merging criterion is not satisfied. The merging stops when all regions are checked. This segmentation approach is described by Algorithm 2.

Algorithm 2 Our graph-based segmentation approach in pseudo-language.

```
Initialization: regard each node as a region and sort edges in a non-decreasing order of weights $w$
for each region do
    Extract the two neighboring nodes $A$ and $B$
    if $A$ and $B$ are different regions then
        compute $W_{\text{m-intra}}(A, B)$ as in Eqn. (4.4)
        if $W_{\text{m-inter}}(A, B) \leq W_{\text{m-intra}}(A, B)$ then
            Join A and B as a new region
            Update the weight of the new region
        end
    end
end
```

4.6.2 Stage 2 of generic semantic region labeling: feature extraction

To train a reliable and robust SVM classifier, in some cases it may be sufficient to use only local features such as color and texture. However, when classes have similar characteristics, complications arise. Adding spatial context in the analysis can be used to tackle such complications. For example, water and sky may have similar color and texture. However, these regions can be often distinguished based on vertical positions, i.e. the sky tends to be at the top of the image and the water at the bottom. Summarizing, we combine the locally calculated pixel-based features and the region-based features to achieve a more reliable region labeling approach.

A. Pixel-based features

Color can be an informative feature for describing a region. In our research we consider three color spaces as candidates for the best representation: HSV, RGB and CIELUV (proposed by Benedek and Szirnyi [112] as the most efficient color space). Besides color, texture features lead to a better classification by analyzing the local neighborhood variation [113]. In this thesis we use the Log-Gabor function proposed by Field [39]. As we depicted in Figure 2.4 in Chapter 2, the filters are constructed in terms of two components: (1) the radial component, which controls the frequency band that the filter responds to, and (2) the angular component, which controls the orientation that the filter responds to. The two components are multiplied together to construct the overall filter. More details of the approach can be found in [35].

B. Spatial Context

When a vertical position is used as feature, the Spatial Context (SC) becomes specific for the region. Our gravity-based model exploits SC to overcome the ambiguities caused by using only color and texture [114]. For each pixel (x, y), we calculate its normalized vertical position $SC_{xy} = x/n$, where x is the row number, y the column number and n is the number of rows belonging to the region.

4.6.3 Stage 3 of generic semantic region labeling: classification

After segmenting the image and extracting the features, we proceed to obtain the labeling results. The labeling is performed by a classification system based on an off-line trained SVM. Here, we present two approaches for region classification, as depicted in Figures 4.7 and 4.8.

A. Classification using the gravity-based model

In this approach, color, texture and SC are used to train the SVM for each region class individually, to achieve unitary-category classification (i.e. an individual SVM is trained for each region type). Later, we randomly sample 100 pixels from a segmented region. The previously trained SVM for the considered class assigns labels to each pixel as positive or negative, depending on the classification results. We calculate the percentage of positive samples in that region. Then, we label the region as belonging to the considered class (e.g. we find the segment depicted by sky), if this percentage of positive samples is higher than an empirically defined threshold. Our fast unitary-category classification is described in the inner part of Algorithm 3.

For multi-category labeling, we assign to each segment one of the 5 labels: sky, vegetation, construction, road, water plus the class "unknown". To this end, we classify each segment by 5 SVMs using our unitary classification (Algorithm 3) and obtain 5 numbers, indicating the percentages of positive pixels for each SVM. Finally, a segment is assigned to a particular class if its percentage is higher than the empirical threshold, which we call T_e from now on. Algorithm 3 illustrates our multi-category classification approach with the unitary algorithm embedded into it. The empirical threshold T_e for each region is set to 50%.

Algorithm 3 Our fast multi-category classification

```
for 5 classes do
    Define the next class type
    for a segmented region do
        Randomly choose 100 samples in this region and use the SVM classifier to label the samples
        Calculate the percentage of positive samples in this region
        if the percentage of positive samples is higher than $T_e$ then
            Set this region is positive
        end
        Compare results to threshold $T_e$
        Label the current region
    end
end
```

B. Classification using the GRS-based model

We define the Global Region Statistics (GRS) as the standard deviation and mean of the region positions. Let us assume that we have M regions of a particular type, for example sky, in the training set of images. For each region, we calculate the mean values μ_k $(k = 1, ..., M)$ of the vertical positions of its pixels, where M is the number of pixels in the region k. We also calculate the standard deviation σ_k of the vertical pixel positions for each region. Then we take minimum and maximum values for all means and standard deviations for this region type:

$$\begin{aligned}\mu_{\min} &= \min(\mu_1, .., \mu_N),\\ \mu_{\max} &= \max(\mu_1, .., \mu_N),\\ \sigma_{\min} &= \min(\sigma_1, .., \sigma_N),\\ \sigma_{\max} &= \max(\sigma_1, .., \sigma_N).\end{aligned} \tag{4.6}$$

where N is the number of samples for each region type. In this way, we obtain intervals for mean and standard deviation for the region position. We assume that the mean value of the vertical pixel positions lies in the interval $(\mu_{\min}, \mu_{\max})$ and standard deviation within $(\sigma_{\min}, \sigma_{\max})$. We compute these intervals for each of the 5 region types described in this study. For a correctly labeled region, the region borders are in the typical interval values for mean and standard deviation of the vertical positions. Therefore, for assigning a label to a region, we check that both following conditions are satisfied: (1) the percentage of positively classified pixels exceeds the threshold T_e; (2) the mean and standard deviation of the vertical positions of the pixels lie in the intervals as discussed above.

The next section will illustrate the experimental results on three datasets: our own dataset, LabelMe [107] and WaterVisie [108]. We also compare our method quantitatively and qualitatively with two state-of-the-art approaches [110][109].

4.7 Experimental results of generic region labeling

Our own dataset consists of a broad range of images from multiple Internet datasets and a personal archive. The images contain 5 region classes (sky, vegetation, road, water, construction) plus the class "unknown". The dataset consists of 255 images: 121 images for training, 134 for testing. All images for training are manually annotated, so that the regions corresponding to the above-described classes are manually delineated. The parameters for graph-based segmentation are as follows. For our applications, $\sigma = 1.4$, $\kappa = 2$ and the minimum region size (S_{min}) equal to 300, are a good choice for complex images. It should be noted that for this approach the accuracy is used for evaluating the results. Regarding the texture extraction in the feature extraction stage, we apply a group of Log-Gabor filters with 1 scale and 8 orientations. For our GRS-based approach, the means and standard deviations are calculated based on 121 images from the training set.

In order to benchmark our approach, we compare our gravity-based model with two state-of-the-art approaches: Bao *et al.* [110] and Millet *et al.* [109].

We have extended the unitary-category classification of Bao *et al.* [110] into multi-category classification and applied contextual information as an additional feature. The rule-based approach proposed by Millet *et al.* [109] relies on preknowledge on the relative spatial positions between regions. More details of the approach proposed by Millet *et al.* [109] can be found in Chapter 5.

We start with a qualitative comparison between our approaches and the approaches proposed by Bao *et al.* [110] and Millet *et al.* [109]. Figure 4.9 illustrates a challenging image from our own dataset along with the comparison results of our gravity-based region labeling with two state-of-the-art approaches, to highlight the differences between the algorithms. This image contains several regions of interest and the color information is quite poor with only small color differences between neighboring regions. Figure 4.9(b) shows the result of our gravity-based region labeling which is similar to the ground truth in Figure 4.9(f). However, our GRS-based approach is successful in labeling the construction region at the bottom of the image, as is shown in Figure 4.9(c). Furthermore, in contrast to our gravity-based results Figure 4.9(d) shows that the approach of Bao *et al.* [110] has labeled part of the construction in the bottom of the image as sky. We can also see that the approach of Millet *et al.* [109] has mistakenly labeled water in the upper part of the vegetation in the image, as is shown in Figure 4.9(e).

To further evaluate the performance of the region labeling algorithms quantitatively, we use the accuracy, which measures how much of the true region is detected by the algorithm [115]. Table 4.2 shows the accuracy comparison for our gravity-based model in three color spaces on 134 images of our dataset. We can observe that the gravity-model in HSV color space results in the highest accuracy. Table 4.3 shows the accuracy comparison for our gravity-based model in HSV color space and GRS-based model approaches, with to Bao's and Millet's algorithms on 134 images of our dataset. We can observe that the gravity-model approach results in a higher accuracy. Our gravity-based approach outperforms the algorithm of Bao *et al.* and our GRS-based model with approximately 2%. Our gravity-based approach also surpasses Millet *et al.* [109] by 3%, while avoiding preset rules which reduce the flexibility of the method. Unlike Millet, our approach does not need to rebuild its status where a new region is added.

We have further benchmarked our gravity-based approach on the LabelMe [107] and the WaterVisie [108] datasets. We have randomly selected 142 images from LabelMe and divided them into 102 training and 40 test images. We have also trained our gravity-based model on 111 frames and tested

Region	Gravity-CIELUV-based	Gravity-RGB-based	Gravity-HSV-based
Sky	93	92	96
Construction	80	87	88
Water	86	89	93
Road	87	88	92
Vegetation	89	87	90
Unknown	94	92	95
Average	87	90	93

Table 4.2: Accuracy (%) comparison for several color spaces.

Region	Millet *et al.* [109]	Bao *et al.* [110]	GRS-based	Gravity-HSV-based
Sky	94	95	96	96
Construction	84	87	88	88
Water	89	89	91	93
Road	89	92	92	92
Vegetation	87	85	85	90
Unknown	99	97	96	95
Average	90	91	91	93

Table 4.3: Accuracy (%) comparison for several semantic labeling algorithms.

it on 16 videos of WaterVisie dataset. Figure 4.10 shows the original images of the datasets and the corresponding results of the gravity-based model. Figures 4.10(b), 4.10(d) and 4.10(f) visualize the results of the gravity model on our own, LabelMe and WaterVisie datasets.

Table 4.4 demonstrates the average of the obtained accuracies for the three studied datasets. The average accuracy over six region classes for the gravity-based labeling approach is 93% for our dataset, 94% for LabelMe and 96% for the WaterVisie dataset. This shows a significant improvement on the LabelMe dataset compared to the 59% reported by Jain *et al.* [116]. We note though that Jain *et al.* [116] aimed at a clearly higher number of semantic region types, which is more difficult.

Approach	Dataset	Coverability Rates (%)
Gravity-based	Our dataset	93
	LabelMe	94
	WaterVisie	96

Table 4.4: Average of accuracy (%) for three datasets.

4.7.1 Complexity-reduced feature selection in generic region labeling approach

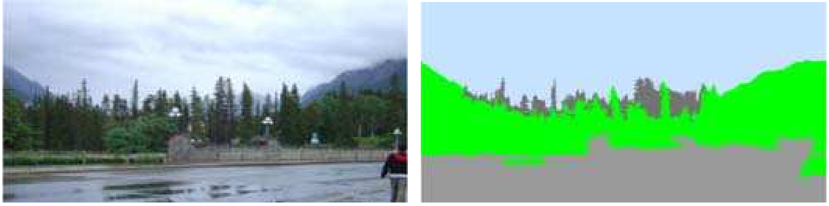
In our classification stage of the gravity-based approach each region is classified by taking 100 randomly sampled pixels in that region and calculating features for each of those pixels. Calculating the HSV and Log-Gabor features takes negligible time compared to the SVM classification. Taking 100 randomly sampled pixels in each region leads to 500 SVM classifications in total for that region. Such number of classifications result in a high computational complexity and may not be optimal for real-time applications. Therefore, in a cooperative work [117] we aimed at improving the computational complexity of our gravity-based region labeling approach. We proposed a different approach which calculates regional features taking the average color and average texture over the complete region. To classify more accurately, next to the averages, also the standard deviations of each of the regional features are taken into account. This means that the previously needed 500 classifications shrink to just 5 for one region. In the next chapter we will investigate the details of computational complexity calculations of the generic region labeling approach.

4.8 Conclusions and discussion

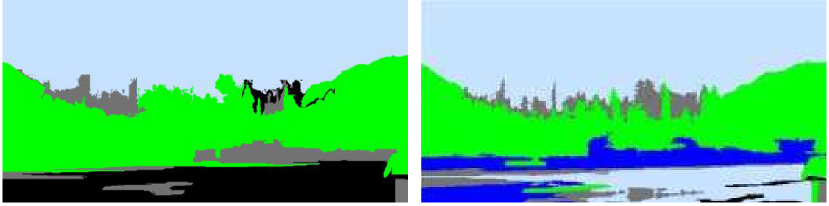
In this chapter, we have presented our research on region labeling for outdoor surveillance applications. Besides our specific region labeling which was discussed in Section 4.3, we have introduced a generic region labeling approach. Our major contribution is introducing a generic framework based on spatial context (in our case vertical position information) for labeling regions. We have introduced two models for generic framework: (1) gravity-based and (2) Global Region Statistics (GRS)-based models. In our gravity-based system, following a segmentation stage, color, texture as well as the vertical position are exploited. For pixel-based features, the HSV color space has been used and a group of Log-Gabor filters have been applied to obtain texture features. These features together with vertical position have been used to train a multiple-SVM classifier. As an alternative, we have presented a system without the gravity-based approach, but with a novel Global Region Statistics (GRS) based model. This model involves the computation of mean and standard deviation of the vertical region positions. The experimental results show that our gravity-based model gives the best results and outperforms our GRS-based approach, the algorithms of Bao *et al.* and Millet *et al.* [109] with 2%, 2% and 3%, respectively. Our gravity-based approach is highly adaptive to new semantic region types because it avoids preset rules which can reduce flexibility. The proposed framework is generic and does not depend on the type of a scene.

Up to now we have considered five semantic region types, but this number can be extended, so that more semantic region types are explored. We expect that such extended approach will be more suitable in this case compared to the specific region labeling.

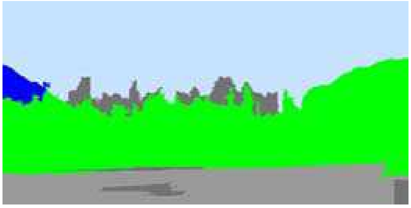
We conclude that our gravity-based generic region labeling approach is more promising compared to our semantic labeling of specific regions because it doesn't need to re-design for each new region type as with specific region labeling. It also outperforms our GRS-based semantic region labeling. Therefore, we will discuss our gravity-based approach in the following chapters for further analysis and evaluation. The computational complexity analysis of the proposed gravity-based region labeling approach will be addressed in Chapter 5 to show that the algorithm also has a low computational complexity. In Chapter 6 we apply our gravity-based region labeling approach to complex surveillance use cases.



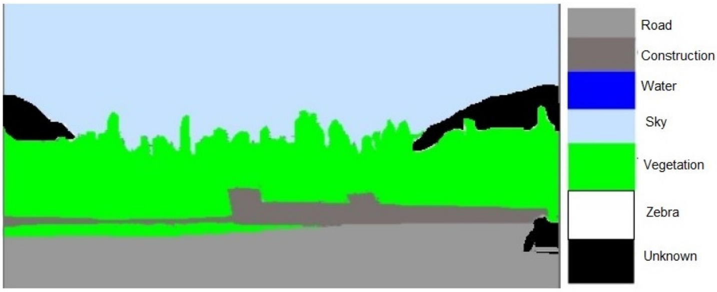
(a) Image from our dataset (b) Our gravity-based region labeling



(c) Proposed GRS-based region labeling (d) Region labeling from Bao *et al*. [110]

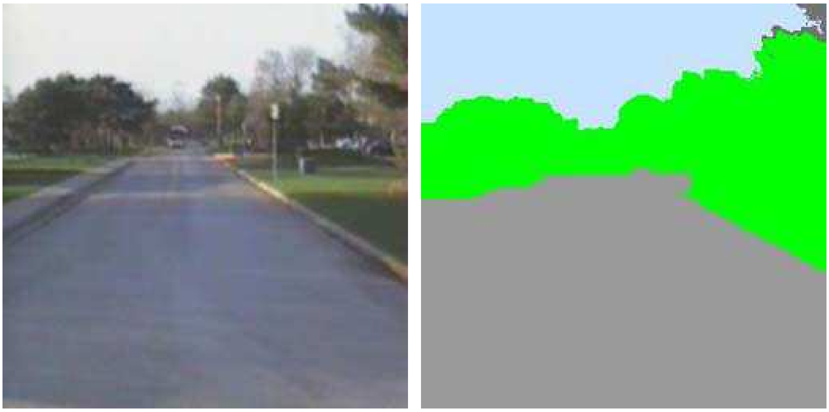


(e) Region labeling from Millet *et al*. [109]



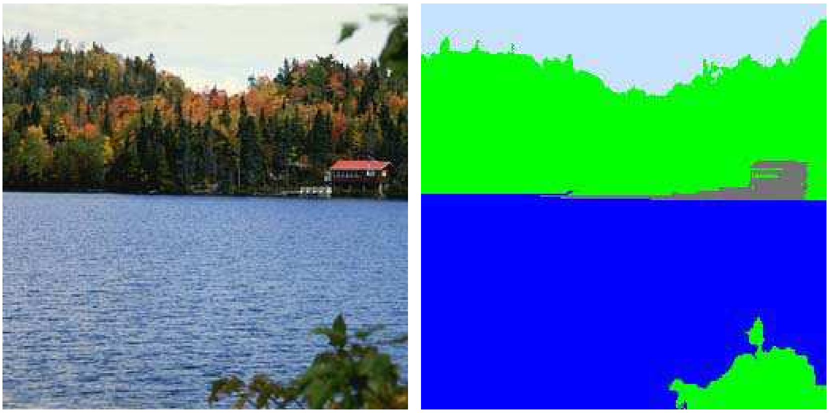
(f) Ground truth of (a)

Figure 4.9: Visual comparison of region labeling approaches.



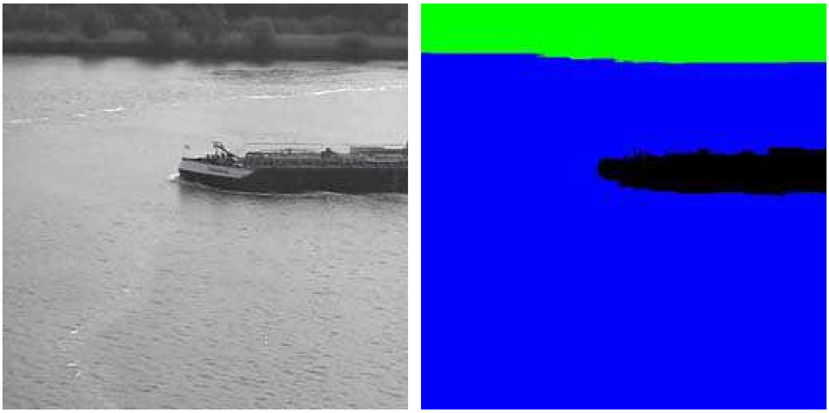
(a) Image from our dataset,

(b) Gravity-based region labeling of (a)



(c) Image from LabelMe [107]

(d) Gravity-based region labeling of (c)



(e) Image from WaterVisie [108]

(f) Gravity-based region labeling of (e)

Figure 4.10: Visual illustration of the gravity model. Light blue, darker blue and green indicate “sky”, “water” and “vegetation”, respectively.

5

Complexity analysis

5.1 Introduction

Chapter 3 has addressed the problem of scene understanding at the region level, i.e. salient region detection. It was shown that the results from our DCT-based approach are better compared to other salient region detection techniques which are based on edge, connected components, Hough transform as well as a prior art approach which is based on color. We have indicated that the DCT-based approach is well-suited for real-time image analysis because it allows parallel processing, since the DCT can be performed for each image block independently and therefore at the same time.

Chapter 4 has discussed the problem of semantic region labeling. In particular, a general framework was introduced which is developed based on color, texture as well as vertical position information as spatial context for performing an automatic semantic labeling task. It was shown that our general framework provides both qualitative and quantitative gains.

In the previous chapters we have noted that our main goal is to develop methods that can be applicable in (near) real-time for embedded surveillance systems. Therefore, the techniques should support both high accuracy and low complexity.

Complexity analysis is designed to estimate computational complexity of an algorithm and therefore allows to compare two algorithms in terms of computational costs. Computational complexity involves of three aspects: (1) time complexity which is defined by execution time, (2) storage space complexity defined by the amount of memory used by an algorithm [118], and (3) memory bandwidth required between computing section and the background memory [77]. A simple way to analyze the complexity of an algorithm is generally calculated by using Big-O notation [119]. A limitation of using Big-O notation, however, is that only the computationally heavy part of the algorithm under analysis is taken into account and it is a very coarse estimate. Thus, this analysis provides insufficient insight into the complexity of different steps of the algorithm under analysis and it is not suited for drawing detailed conclusions. Albers *et al.* [77] introduce an approach that takes into account the three aspects mentioned above, i.e. time, space as well as required bandwidth, with the aim to gain more insight in the dynamic and runtime complexity. Their complexity analysis method is based on counting native Digital signal processing (DSP) operations with a basic RISC CPU as a reference model. Furthermore, Albers *et al.* [77] assume a processor architecture in the reference model, i.e. a processor core connected to a background memory. In this thesis we adopt an algorithm from Albers *et al.* [77] since it provides sufficient insight into the complexity of different steps of the algorithm under analysis.

In this chapter, we provide the computational complexity analysis of our proposed approaches for salient region detection and generic semantic region labeling which were addressed in Chapter 3 and Chapter 4, respectively. By analyzing the computational complexity of our approaches and comparing them with the prior art algorithms, we address the question whether or not our approaches are suitable for (near) real-time applications.

The structure of this chapter is as follows. Section 5.2 reviews the system that we are going to analyze with respect to complexity. Section 5.3 provides an introduction to the method we use for analyzing computational complexity of algorithms. Section 5.4 presents complexity analysis of our salient region detection approach. Section 5.5 provides complexity analysis of our generic semantic region labeling technique. Discussion and conclusions related to the computational complexity of the proposed approaches are presented in Section 5.6.

5.2 Context-based system

In video surveillance systems, videos acquired by cameras are commonly stored in large databases for later retrieval. Scene information which is extracted by video analysis algorithms is stored along with the video data as metadata. Large video databases should facilitate user-friendly access and browsing. So far in this thesis, we have discussed that it is important for surveillance systems to exploit analysis at different semantic levels. For example, such analysis can assist in quickly classifying the images into outdoor and indoor images. In another example, it can be used in querying databases, e.g., for finding greenery or salient regions. Analysis at semantic levels requires that the differences between the pixel, object and scene levels are bridged in a joint analysis tool. Besides these aspects which address the accuracy of the systems, the ideal video analysis system should provide high processing efficiency, achieving (near) real-time operation.

Figure 5.1 illustrates a schematic view of our surveillance video analysis algorithms with high accuracy and low computational complexity. Our surveillance video analysis system consists of two main components: (1) the DCT-based salient region detection technique presented in Chapter 3 and (2) our semantic region labeling algorithm proposed in Chapter 4. In addition to the key functions for computations for each component, which we have discussed in Chapters 4 and 3, also memory storage is required. The memory storage requirements can be allocated for current input pixels, Look-Up-Table (LUT) buffers and outputs. In the following sections, we study the computational complexity of our proposed approaches.

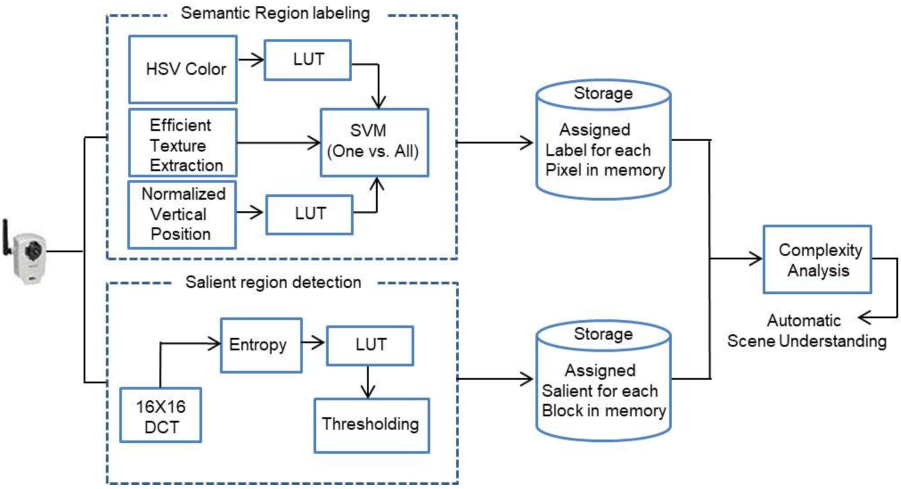

Semantic Region labeling
HSV Color
LUT
Efficient Texture Extraction
SVM (One vs. All)
Normalized Vertical Position
LUT
Storage
Assigned Label for each Pixel in memory
Complexity Analysis
Automatic Scene Understanding
Salient region detection
Entropy
LUT
16X16 DCT
Thresholding
Storage
Assigned Salient for each Block in memory

Figure 5.1: Schematic view of context-based surveillance image and video analysis algorithms with high accuracy and low computational complexity.

5.3 Computational complexity estimation method

A direct way to define the complexity of an algorithm can be the analysis of the amount of time and memory the algorithm needs for execution and providing the result for storing [120]. Here, we adopt a computational complexity estimation method from Albers *et al.* [77]. In this method the following three elements are addressed for analyzing computational complexity of an algorithm: (1) execution time, (2) storage and (3) memory bandwidth.

A known metric for estimating the computational complexity is Operations Per Frame (OPF), Mega Operations Per Frame (MOPF) or Per second (MOPS) [77]. These metrics are based on the following aspects:

- Native DSP computations
- Memory storage and bandwidth
- Processing model

Native DSP computations. In this metric only native operations like multiplications, additions, etc. are considered. For example, consider a simple linear equation, $y = ax + bz$ where a, b are constants and x, y and z are data. This requires 4 data loads, 2 multiplications, 1 add, 1 data storage, thus 8 operations.

Memory storage and bandwidth. By incorporating data storing and loads, also an indication of the memory usage and bandwidth is obtained. In this model, address computations are omitted (this is existing parallelism in processor hardware). Furthermore, the data traffic from cache to memory is omitted (cache hierarchy is present and working properly). In [77], caching was incorporated, but skipped here for simplicity.

Processing model. Here, a simple RISC CPU processor is assumed with sufficient background memory for video data storage, but with a limited set of registers. RISC processors only use simple instructions that can be executed within one clock cycle. Processors consist of two parts, the arithmetic/logic unit (ALU) and the control unit. The former performs arithmetic and logical operations, the latter controls the flow of operations. In addition to the processor there is memory. The core of the data processing unit consisting of ALU and registers is shown in Figure 5.2. Evidently, data cycle from registers into the ALU, where an operation is performed, and the result is transferred back into the background memory.

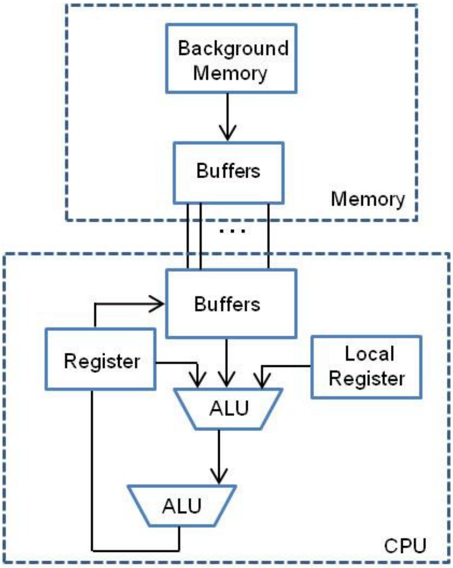


Figure 5.2: Processor with the RISC core, ALU and registers.

5.4 Complexity analysis of salient region detection approaches

This section discusses the execution time for our DCT-based technique as well as for different salient region detection approaches which are based on edge, connected components, and the Hough transform. Furthermore, details of computational complexity calculations of our DCT-based salient region detection approach are discussed. Additionally, we provide computational complexity calculations of the approach proposed by Rahtu *et al.* [90].

5.4.1 Execution time

In previous studies, e.g. in [77], it has been demonstrated that the computational complexity greatly impacts the execution time. Therefore, higher computational complexity of algorithms implicitly indicates a higher execution time. It is however noted that the execution time also depends on the implementation of the algorithms. For example, a MATLAB implementation of different salient region detection approaches which we have discussed in Chapter 3 produces the profile shown in Table 5.1. These results are taken on a system comprised of CPU core i7 2.4-3.4 GHz Quad core processor supporting hyperthreading, 8 GB DDR RAM running at 1600 MHz on Windows 8.1 Pro 64-bit. Table 5.1 shows that the Edge-based approach has a lower execution time and Hough-based technique is the most time consuming approach.

In Table 5.1 we note that the DCT-based approach executes faster than the Hough-based and is slower than the edge-based and connected component-based techniques. However, from Chapter 3 we know that the DCT-based approach outperforms the edge-based and connected component-based techniques in terms of precision and recall rates. Therefore, our DCT-based salient region detection approach is adopted as the candidate for computational complexity analysis.

Approach	Execution time (milliseconds)
Edge-based	63
Connected component-based	105
DCT-based	150
Hough-based	225

Table 5.1: Average computational time of salient region detection methods per frame, calculated over 5454 frames of the Caltech Pedestrian Detection Benchmark [89].

5.4.2 Complexity estimation of our DCT-based salient region detection

Here we explain the details of the complexity calculation of the proposed DCT-based salient region detection illustrated in Figure 5.3. Based on our calculations which will be presented later in this section, the most computationally heavy step of our approach is applying a 2D DCT to each RGB channel. To reduce computations, we consider an efficient and fast forward 2D DCT algorithm proposed by Wang *et al.* [121]. Figure 5.4 shows a signal flow graph proposed by Wang *et al.* for $N = 16$. Inputs are 16 pixels, and they are processed with 9-stage butterfly operations. A butterfly is the transform of 2 samples $(a, b) \mapsto (a + b, a - b)$ [122].

The amount of reads, adds and storages required by our 9-stage butterfly operations is depicted in Table 5.2. In this table we can see that the complexity

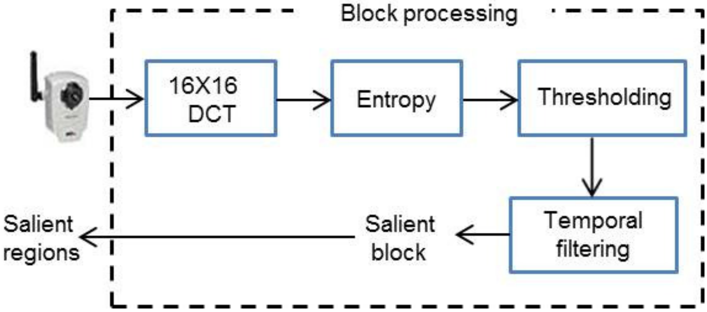


Figure 5.3: Proposed DCT-based salient region detection on the red component from an RGB image.

requirements of our fast forward 2D DCT algorithm per block are 273 operations. Therefore, we need $266 \times 4,800$ or 1.276 MOPF (4,800 is the total number of blocks per channel). The entropy of the DCT coefficients of each 16×16 block requires 16 reads for probability distribution of each DCT coefficient, which we have saved in a lookup table off-line. Additionally, 16 reads for the logarithm of the probability, 16 multiply accumulations and 1 data storage in the memory are needed.

Table 5.3 illustrates the influence of the block size on computational com-

Algorithm stages	Operations/Frame
1	16R + 16A + 16W
2	8R + 8A+ 8W
3	5R + 4A + 4W
4	12R + 8A + 8W
5	14E + 14A + 14W
6	20R + 8A +8W
7	8R + 8A +8W
8	20R + 8A + 8W
9	16M + 16W
Total (OPF)	273

Table 5.2: Computational complexity of salient region detection (R, A and W refer to Read, Add and Write).

plexity calculations of our salient region detection using a typical highway

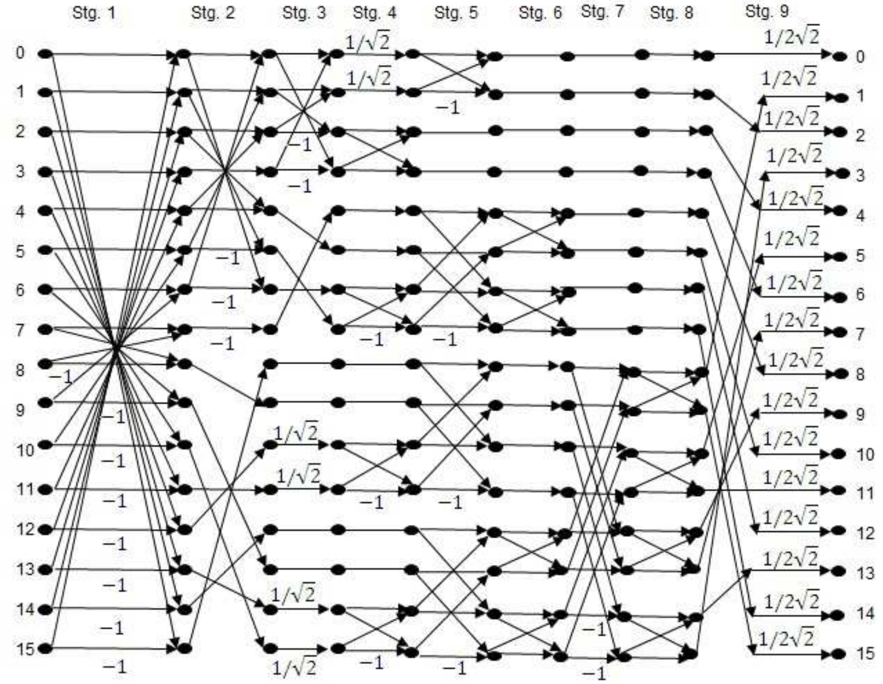


Figure 5.4: Butterfly diagram proposed by Wang *et al*. [121] (stg. refers to stage).

video sequence. From Table 5.3 we can observe that although the computational complexity of our salient region detection approach for 8×8 blocks requires less computations per block compared to 16×16 block size, the increased number of blocks from 4,800 to 19,200 increases the total number of computations. Table 5.3 shows that the total operation count for all 4 stages on 16×16 block size is 5.22 MOPF. This sequence consists of 40 frames with a resolution of $1,280 \times 960$ pixels. For applications requiring full-HD resolution video, the herein presented calculations should be multiplied by 16 ($16 \times 5.22 = 83.52$ MOPF).

Algorithm steps	MOP of 8x8	MOP of 16x16
2D DCT	5.8752	3.369
Entropy	2.016	0.9648
Thresholding	1.83	0.45
Temporal filtering	0.00576	0.00144
Total	9.72	5.22

Table 5.3: DCT-based salient region detection complexity analysis per frame on the highway dataset (frame size is $1,280 \times 960$ pixels).

5.4.3 Complexity estimation of Rahtu's saliency detector

This section provides the details of the complexity calculation of the saliency detection approach proposed by Rahtu *et al.* [90]. Table 5.4 illustrates the complexity analysis of salient region detection by Rahtu *et al.* [90] for each step of the algorithm.

This algorithm is based on searching image segments whose intensity values are described by the intensity distribution of the object, compared to the distribution of the surrounding area. The algorithm of Rahtu consists of the four following steps, as depicted in Table 5.4: convert RGB color space to Lab color space, integral histogram, Gaussian filter, and maximum of histogram for each window. The most computationally heavy stage of Rahtu's approach is the RGB-to-Lab conversion step. This conversion step involves another two steps of $RGB2CIE^{*}XYZ$ and $CIE^{*}XYZ2Lab$ [123]. Here, we

Algorithm steps	MOPF
Convert RGB to Lab color space	50
Integral histogram	3.69
Gaussian filter	0.028
Maximum of histogram per window	2.48
Total	56.19

Table 5.4: Complexity analysis of salient region detection by Rahtu *et al.* [90] on the highway dataset (frame size is $1,280 \times 960$ pixels).

explain the step of $RGB2CIE^{*}XYZ$ which can be derived using:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4125 & 0.3576 & 0.1804 \\ 0.2127 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9502 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}, \qquad (5.1)$$

where X, Y and Z are color components in XYZ space and R, G and B are color components in RGB space. This conversion requires 3 reads for R, G and B, 9 multiplies, 6 adds and 3 data stores of X, Y and Z in the memory. Therefore, the conversion involves 21 operations/sample and these operations are applied to every pixel. Complexity calculations for the other stages of this approach are obtained in a similar way.

The computational complexity of the salient region detection approach of Rahtu *et al.* [90] equals 56.19 MOPF (see Table 5.4). Table 5.3 shows that the total operation count for all 4 stages of our salient region detection is 5.22 MOPF when we choose 16×16 block size for processing. This comparison shows that our approach outperforms Rahtu's algorithm with 10 times lower complexity, while it has been earlier reported in Chapter 3 that the accuracy of our salient region detection is 22% higher than that of Rahtu *et al.* [90].

5.5 Complexity analysis of semantic region labeling approaches

This section begins with the computational complexity estimation of our gravity-based semantic region labeling approach. Furthermore, the execution time for different stages of our approach is discussed. Additionally, we provide computational complexity calculations of a similar approach proposed by Millet *et al.* [109].

5.5.1 Complexity estimation of gravity-based semantic region labeling

In this section, the details of the complexity calculations of the region segmentation stage are presented. These details are provided because this stage involves more complicated calculations compared to the other stages. Computational complexity estimation of the feature extraction and classification stages have been obtained in the same way and reported in this section as well.

In the region segmentation stage, a graph-based segmentation approach is applied, which is motivated by the concept that pixels within one region are closer in color space than pixels from other regions. Based on this, a threshold is defined depending on the size of the region, where it is assumed that larger regions should have larger tolerances for color deviations.

The algorithm steps of our graph-based segmentation approach (see Chapter 4) are summarized in the left column of Table 5.5. Our graph-based segmentation stage has five steps: 1) Gaussian filter, 2) finding different regions, 3) weight calculation, 4) merging comparison, 5) weight updating, as indicated in the left column of Table 5.5. Let us discuss the computational complexity estimation of these steps.

Step1: *Gaussian filtering*. In the graph-based segmentation, the input image is first filtered with a Gaussian kernel. It is proven that an efficient way to convolve an image with a 2D Gaussian kernel is to choose a separable filter and then split the 2D Gaussian kernel into concatenation of two 1D kernels. The 1D convolution is then specified by $(I * K)(x, y) = \sum_{j=1} K(j) \cdot I(x, y - j + 1)$, where $I(x, y)$ denotes the intensity of the current pixel (x, y) and $K(j)$ is the 1D kernel. Since the kernel size is 1×2, the filter requires 2 coefficient reads for the 1D kernel $K(j)$, 2 additions for $y - j + 1$, 2 image pixel reads for $I(x, y - j + 1)$, 2 multiply accumulations for $K(j) \cdot I(x, y - j + 1)$ and 1 data store for moving the result in the memory. Later, this computation is repeated vertically and further for each color component per pixel.

After smoothing the input image with a Gaussian kernel, the segmentation algorithm requires the steps of finding regions, weight calculation, merging comparison and updating of the weight (left column of Table 5.5).

Step 2: *Finding regions*. The step of finding regions (if $\mathrm{Region_A} \neq \mathrm{Region_B}$), involves 2 reads for $\mathrm{Region_A}$ and $\mathrm{Region_B}$ and 1 comparison. This computation is pixel-based.

Step 3: *Weight calculation*. The step of weight calculation W_A involves 1 data read of inter-region weight W_A from a look-up table. It should be noted that the edge weights according to Eqn. (4.3) in Chapter 4 and the maximum edge weights for each pixel are calculated and saved in a lookup table in an off-line calculation and they are updated later depending on the outcome of the condition. Prior to updating, a merging step is first investigated.

Step 4: *Merging comparison*. The step of merging comparison, involves calculating $W_{m-inter}(A, B) = W_{A,B}$, where the subscripts "$A, B$" denote taking regions A and B together. Weight $W_{A,B}$ requires 2 reads for W_A and W_B, 1 comparison to achieve the maximum of W_A and W_B and 1 data store of $W_{A,B}$ in the memory. In addition, this step according to Eqn. (4.4) requires 2 reads for $|A|$ and $|B|$, 1 read for κ, 2 multiplications for $\kappa/|A|$ and $\kappa/|B|$, 2 adds and finally 1 comparison. If the criterion is satisfied, then the graph-based segmentation algorithm continues to the next step, which is "Updating the weight of the new region" as depicted in Algorithm 2 in Chapter 4.

Step 5: *Updating weight*. Finally, we return to updating the weights. This stage is based on the merging comparison result. If $W_{m-inter}(A, B)$ is smaller than or equal to $W_{m-intra}(A, B)$, then the two regions A and B are merged and $W_{A,B}$ is updated for two regions.

Except for the Gaussian filter (a pre-processing step), all above-mentioned steps of the region labeling are repeated by the height of the graph, which equals $n \times \log(n)$, where n denotes the number of pixels covering the height.

Table 5.5 shows the estimation results of the computational complexity of the semantic region labeling on the "LableMe" dataset [107] with the frame size of 256×256 pixels. In this study, this public dataset has been selected for scientific reference only. For applications requiring full-HD resolution video, the herein presented calculations should be multiplied by about $32(14,933,248 \times N_R$ + 267,911,648) and the frame rate, so that MOPF counts become GOPS with nearly the same numbers, proving that region labelling is feasible for real practical systems.

Table 5.5 shows that the total operation count for all 3 stages is $466,664 \times N_R$+ 8,372,239 OPF (N_R is the number of randomly selected pixels).

A. Execution time

As mentioned above, a higher computational complexity of algorithms implicitly indicates a higher execution time. The execution time also depends on the platform and programming environment on which the algorithms are implemented. For example, an implementation of our generic semantic region labeling approach in C++ using the OpenCV library produces the profile as shown in Table 5.6. This profile was composed with the Visual Studio profiling tool and is therefore not platform independent, but gives nevertheless insight in the bottlenecks of the code.

All results shown in Table 5.6 are taken on a system comprised of CPU core i7 2.4-3.4GHz Quad core processor supporting hyperthreading, 8 GB DDR RAM running at 1600 MHz on Windows 8.1 Pro 64-bit.

From Table 5.6 it is observed that the CPU time required for the classification stage, 78.4%, is completely out of proportion. The reason for this high cost in CPU-time is that the SVMs classify 100 pixels for each region. Each pixel is in practice classified by 5 SVMs because we have 5 region types. This amounts to 500 SVMs classifications for 5 regions. The segmentation of the image also takes 15.6% of the total CPU time and together both stages account for over 90% of the total CPU time. Given these high CPU time percentages for segmentation and classification stages, we hypothesized that reducing the number of pixels increases the efficiency of the approach in terms of CPU time. To test this hypothesis, as described in Chapter 4, in a cooperative work [117] we took into account the average and the standard deviations of each of the regional features. This means that the previous 500 classifications lower to just 5. Table 5.7 shows that the execution time is reduced with the improved implementation. This improvement provides a clear step forward towards (near) real-time implementation. It is noted that all execution-time experiments provided here are conducted on the 481×321 images of our dataset and it is known that the execution time of segmentation and Gabor filtering are heavily relying on the image dimensions, so that a lower resolution input will result in faster classification.

In addition to the analysis of the execution time provided above, it is further noted that the execution time can be improved using the OpenMP (Open Multi-Processing) interface. For example, in [117] we showed that when using OpenMP the execution time of 272 ms described in Table 5.7 is reduced to 125 ms (i.e., a 54% decrease in the execution time). Furthermore, it should be noted that the execution time also depends on the hardware specifications of the platform on which the algorithm is executed.

5.5.2 Complexity estimation of Millet's approach

In order to benchmark our approach, we compare our semantic region labeling with a similar approach proposed by Millet *et al.* [109]. It is noted that at global level, the rule-based algorithm proposed by Millet *et al.* [109] is similar to our approach and it also involves 3 stages: segmentation, feature extraction and classification. However, there are several differences. The major difference between the two approaches is that in Millet's approach, the SVM classifier is not trained for spatial context. Instead, in order to obtain the relative spatial positions between regions, the following steps are defined: 1) angle computation, 2) normalized histogram, 3) confidence function and 4) consistency function. Our approach further differs from Millet *et al.* [109] in the classification stage. Figure 5.5 depicts the different stages of Millet's approach. It should be noted that for the common stages between our approach and Millet's work, we use the same techniques to establish a fair comparison.

We now discuss in detail the computational complexity estimations of spatial context extraction proposed by Millet *et al.* [109].

Step 1: *angle computation*. The first step of spatial context extraction, i.e. computing an angle between each two regions calculated by inverse tangent (atan2) , $\text{atan2}((Y_2 - Y_1), (X_2 - X_1)) \times 180/\pi$ (where (X_1, Y_1) and (X_2, Y_2) de-

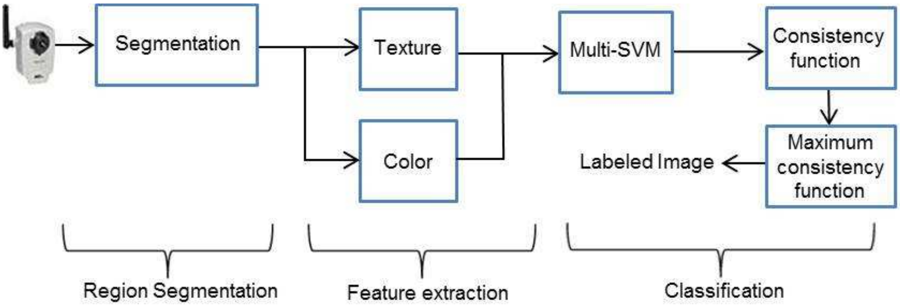


Figure 5.5: Semantic region labeling approach of Millet *et al.* [109].

note two pairs from image coordinates for Region_1 and Region_2). The term atan2 stands for the arctangent function. This step involves 4 reads for the pixel position, 2 additions to obtain $(Y_2 - Y_1)$ and $(X_2 - X_1)$, 1 read for atan (which is saved in a lookup table in an off-line calculation), atan $\times$ $180/\pi$ requires 2 multiplications and finally 1 data store in the memory. It should be noted that the histogram is also saved in a lookup table in an off-line calculation.

Step 2: *histogram normalization*. The next step, normalized histogram, involves 1 read from a lookup table, then 1 multiplication to obtain normalized value and 1 data store in the memory. The steps of angle computations and normalized histogram repeat the operations for 500 pixels and 12 times for each segment.

Step 3: *confidence function calculation*. The next step is calculating the confidence function between each two regions. For example, if h is the angle histogram "Region_2 is right of Region_1" is verified with the confidence R_{right} defined by: $R_{right} = \sum_{\theta=-90}^{\theta=+90} h(\theta)\cos^2(\theta)$. This gives the relation of Region_2 regarding Region_1; an angle of 0 radian means that Region_2 is right of Region_1, and that Region_1 is left of Region_2. This step involves 180 reads to achieve θ, 180 reads for histogram, 180 operations to obtain $\cos^2(\theta)$ which is also saved in a lookup table. Furthermore, the step requires 180 multiply accumulations and 1 data store in the memory. The overall step repeats 4 times for each spatial relationship (right, left, above and below).

fand result: *consistency function calculation*. The next step is consistency function calculation. Given N regions Region_i in the image that may be backgrounds according to the SVMs, a consistency formula is calculated for each possible case. Parameter B_i is a background attributed to the Region_i. An example case can be: B_1 = sky, B_2 = unknown,B_3 = water, meaning that "Region_1

is sky, Region$_2$ is not a background, and Region$_3$ is water". Millet *et al.* [109] propose using the following formula:

$$C(R_i(B_i), R_j(B_j)) = P(B_i) \times P(B_j) \times (R_i \Re R_j) \times Eval(R_i \Re R_j), \quad (5.2)$$

where $P(B_i)$ is the probability of detection of the background B_i for the region R_i returned by the SVM. Symbol $\Re$ denotes a spatial relationship between two backgrounds. This consistency function calculation step involves 2 reads for $P(B_i)$ and $P(B_j)$, 3 reads and 1 data store of $(R_i \Re R_j)$ in the memory. Here, the operation $\Re$ is a spatial relationship between two backgrounds, where $(R_i \Re R_j)$ is the degree of confidence for the spatial relationship $\Re$ between the regions R_i and R_j. If for example region R_i is 80% above region R_j, 10% below and 10% right of it, then $(R_i \text{ above } R_j) = 0.8$. A knowledge-based function is defined as Eval$(B_i \Re B_j)$. It returns a value representing whether a relation $(B_i \Re B_j)$ between two backgrounds is in agreement with the rules (+1), in contradiction with them (-1), or not represented by them (0). This step further requires 1 read for Eval$(R_i \Re R_j)$, 3 multiplications to calculate $C(R_i(B_i), R_j(B_j)) = P(B_i) \times P(B_j) \times (R_i \Re R_j) \times$ Eval$(R_i \Re R_j)$ and 1 data store in the memory. The step repeats 4 times for each label. The step of the maximum consistency function requires 1 read for the consistency function, 3 comparisons to obtain the maximum value over 3 values and 1 data store in the memory. The resulting numbers can be read from Table complexityMillet.

As we discussed earlier, our approach further differs from Millet *et al.* [109] which will be addressed in the following. The left column of Table 5.8 illustrates the steps of the differences between our approach and Millet *et al.* Since in Millet's approach the SVM is not trained by the spatial context, the number of support vectors becomes larger, as the number of support vectors relies on the number of features used to train SVM. In our approach, the average number of support vectors over 5 regions is 25,923 while for Millet this number equals to 33,182. In the kernel function $K(X_s, z) = e^{-||X_s - z||^2/2\sigma^2}$, the distance function $||X_s - z||^2$ represents the squared Euclidean distance between the support vectors, i.e. X_s and feature vectors, i.e. z. We save the complexity calculations of the kernel function in a lookup table in an off-line operation for our approach as well as for Millet's approach. Thus, the kernel function involves 2 reads for X_s, z and 1 read for $K(X_s, z)$ from the lookup table, which is repeated over the number of support vectors (SV), i.e. $3 \times 25,923$ (in our approach).

The kernel function stage, $\sum_{i \in SV} \alpha_i \cdot K(X_s, z) + b$, is repeated for each of the N_R randomly selected pixels, within each segment for each region type. Therefore, the total number of calculations for the kernel stage in our approach becomes $3 \times 25,923 \times N_R \times 12 \times 5 + 4 \times N_R + 15$ which amounts to 4,666,144 $\times$ N_R+15 OPF while Millet's approach requires 3 $\times$ 33,182 $\times$ N_R $\times$ 12 $\times$ 5 +108, resulting into 5,972,760 $\times$ N_R+108 OPF.

The computational complexity of the semantic region labeling approach of Millet *et al.* [109] for all the stages together equals $5,972,760 \times N_R + 8,467,471$ OPF. Table 5.5 shows that the total operation count for all 3 stages is 466,664 $\times$ N_R + 8,372,239 OPF. This comparison shows that our approach outperforms Millet's algorithm with a 24% lower complexity (for $N_R = 100$), while it is reported in Chapter 4 that the region labelling accuracy of our algorithm is at the same time higher than that of Millet *et al.* [109] (93% vs. 90%).

5.6 Discussion and conclusions

In this chapter, we have discussed a context-based surveillance image and video analysis system, which consists of two main components: (1) our DCT-based salient region detection technique presented in Chapter 3 and (2) our semantic region labeling algorithm discussed in Chapter 4. We have presented and analyzed the computational complexity of these main components. In our analysis, we have estimated the complexity of the developed algorithms, to prove that our algorithms are not only offering accurate region analysis, but also execute with low complexity, to support application in (near) real-time and embedded systems. The applied complexity analysis method is based on counting native DSP operations and memory storage actions with a basic RISC CPU as a reference model.

We have evaluated our salient region detection techniques based on various features, i.e., sum of edge pixels using a Sobel edge detector, number of connected components based on the intensity values of neighboring pixels, the number of straight lines found by the Hough transform and the entropy of the frequency-domain features of a DCT, in terms of execution time. The execution time analysis and the results from Chapters 3 we conclude that our DCT-based approach is an efficient technique, while it maintains a high accuracy compared to the other approaches we have proposed. The computational complexity analysis shows that our DCT-based salient region detection outperforms Rahtu's algorithm with about 10 times lower complexity. The complexity analysis shows that in Rahtu's algorithm, the step of converting RGB-to-Lab color space has the highest contribution (88%) in the total complexity calculation (56.19 MOPF) of the approach. We realize that the conversion RGB-to-Lab color space can be implemented in an FPGA-based processing platform and thereby it can be eliminated from the analysis processing. However, even in this case, the complexity of the remaining algorithm steps is 6.198 MOPF which is still 18% higher than the total complexity calculations of our approach.

Furthermore, we have estimated the computational complexity calculations of each step of our gravity-based region labeling approach. We have compared the computational complexity calculations of our gravity-based region labeling approach with the approach proposed by Millet *et al.* [109] which is similar to our approach and involves combining color, texture and spatial context as feature information in a learning-based fashion, using trained SVMs. Compared to the Millet's approach, in our algorithm we train the semantic region labeling algorithm with normalized positions while Millet's approach relies on pre-knowledge on the relative spatial positions between regions. The rule-based approach increases the number of features used to train SVM classification and thereby increases the number of support vectors, which in turn increases the complexity of SVM classification. In our approach the average number of support vectors is 28% lower than that of Millet's approach and consequently, the complexity of our classification stage is decreased by $1,306,616 \times N_R + 93$ OPF. Furthermore, in addition to the classification stage, the thresholding stage of Millet's approach requires extra processing steps: computing angle, normalize histogram, confidence function, consistency function. These extra steps add additional operation calculations of 0.065 MOPF (for $N_R = 100$) compared to our step.

In summary, it can be concluded that both approaches presented and discussed in this study are suitable for real-world applications in surveillance videos. Fortunately, the application of the presented algorithms goes beyond surveillance and may also be employed in advanced video systems to perform real-time content-dependent quality optimization. Initial attempts of such an approach can already be found in newly presented high-quality TV displays.

Table 5.5: Computational complexity of our gravity-base semantic region labeling.

Algorithm steps	Formula	Number of pixels	Operations/Frame
Stage 1: Region segmentation: 1) Preprocessing: Gaussian filter 2) Finding regions 3) Weight calculation 4) Merging comparison 5) Updating weight	$(I \star K)_(x) = \sum g(x).f(x-y+1)$ If Region$_A \neq$ Region$_B$ W_A $W_{m-inter}(A,B) \leq W_{m-intra}(A,B)$ $W_{m-inter}(A,B)$	256×256 pixels	$9 \times 3 \times 2 \times 256 \times 256$ $+3 \times 256 \times \log 256$ $+1 \times 256 \times \log 256$ $+12 \times 256 \times \log 256$ $+256 \times \log 256$
Stage 2: Color: RGB to HSV	$V = \max(R,G,B)$ $S = V - \min(R,G,B)$ $H = (G-B)/S\, if\, V = R$ $H = 2 + (B-R)/S\, if\, V = G$ $H = 4 + (R-G)/S\, if\, V = B$	256×256 pixels	$38 \times 256 \times 256$
Stage 2: Texture: 1) RGB2GRY 2) Filter construction 3) Multiplication 4) Inverse transform	$0.299.R + 0.587.G + 0.114.B$ $F(f) =$ Radial component × angular component $R =$ imageFFT $\star F(f)$ Ifft2(R)	256×256 pixels	$9 \times 256 \times 256$ $+2 \times 7168+$ $26 \times 256 \times 256$
Stage 2: Spatial context: Normalized position	Vertical position/image height	randomly selected pixels	$N_R \times 5 \times 2$
Stage 3: Classification: 1) Stage kernel function 2) Probability 3) Maximum probability 4) Thresholding	$\sum_{s \in SV} \alpha_s \times K(X_s, z) + b$ % positive samples Maximum Maximum > Threshold	each 12 segments and 5 regions randomly selected pixels randomly selected pixels each 12 segments	$3 \times 25,923 \times N_R \times 12 \times 5$ $2N_R + 2$ $2N_R + 1$ 1×12
Total (OPF)			466,664N_R + 8,372,239 OPF

Algorithm stage	CPU time (%)
SVM classification	78.4
Image segmentation	15.6
Gabor filter bank	1.0
Post processing results	0.1
Miscellaneous code	4.9
Total execution time	4,400 ms

Table 5.6: Profile statistics for our generic semantic region labeling on 49 images of our dataset with 481×321 resolution.

Approach	Execution time (milliseconds)
Gravity model	4,400
Faster feature selection [117]	272

Table 5.7: Execution time comparison on 49 images of our dataset with a resolution of 481×321 pixels.

Table 5.8: Computational complexity of the difference between our semantic region labeling and of Millet *et al*. [109].

Difference steps of present and Millet *et al*. [109] approaches	Formula		Operations/Frame
Stage 2: Spatial context:			
1) Angle computation	atan2$((Y_2 - Y_1), (X_2 - X_1)) \times 180/\pi$		$12 \times 500 \times$ ‘10
2) Normalized histogram	Saved in a look up table		3
3) Confidence function	$R_{right} = \sum_{\theta=-\pi/2}^{\theta=+\pi/2} h(\theta) \times cos^2(\theta)$		$(180 + 180 + 180 + 180 + 1) \times 4 \times 12$
4) Consistency function	$C(R_i(B_i), R_j(B_j)) = P(B_i) \times P(B_j) \times (R_i \Re R_j) \times Eval(R_i \Re R_j)$		44×12
Stage 3: Classification:			
1) Stage kernel function	$\sum_{i \in SV} \alpha_i \times K(X_i, z) + b$		$3 \times 33182 \times N_R \times 12 \times 5$
2) Thresholding	Max(Consistency function)		5×12
3) Labeling	Finding corresponding index as label		4×12
Total (OPF)			5,972,760N_R + 95,247 OPF

6

Applications

6.1 Introduction

At the start of this thesis Chapter 2, has introduced several levels of scene understanding, i.e., pixel, region/object, context and scene. It has been also acknowledged that region analysis can have two general applications: (1) to detect an arbitrary region in an image for general purposes, or (2) to serve as contextual information when it is inserted as a processing block in another larger algorithm or an application. Chapters 3, 4 and 5 have shown that the proposed approaches for extracting information at the region level, i.e. salient region detection and semantic region labeling approaches quantitatively and qualitatively outperform other approaches in accuracy, while operating at several times lower complexity.

The aim of this chapter is to present several use cases where the techniques developed in this thesis were used to improve and assist in a better semantic interpretation of events occurring in a monitored space. In particular, this chapter intends to show that using contextual information in traffic surveillance applications increases the reliability of the detection of objects (e.g. ships in harbor and cars in street surveillance videos) which helps to achieve a high level of surveillance scene understanding. Furthermore, this chapter attempts to motivate that using contextual information enables the automated analysis of complicated traffic surveillance scenarios (e.g. abnormal traffic actions) that were previously not possible using conventional object classification techniques. We validate these ideas by presenting four traffic surveillance use cases, in which the contextual information is exploited for surveillance scene understanding, as follows.

- *Detection of moving ship in harbor surveillance*: in this use case, our semantic region labeling approach introduced in Chapter 4 is combined with motion context information to detect moving ships in port surveillance videos.
- *Recognition of traffic action*: in this use case, our semantic region labeling algorithm proposed in Chapter 4 and automatic traffic-sign information [3] are combined to recognize actions in traffic surveillance video.
- *Detection of moving cars in traffic surveillance*: in this scenario, two use cases are presented to detect moving cars in traffic surveillance videos.
  - In the first use case, our semantic region labeling algorithm introduced in Chapter 4 is again combined with motion information.
  - Additionally, our DCT-based salient region detection technique from Chapter 3 is again combined with motion information.
- *Fast abnormal event detection*: in this use case, a novel block-based approach is developed based on analyzing the pixel-based motion context, as an alternative for the conventional object-based approach.

The first use case addresses an automatic video-based ship detection, which is an important research area for security control in port regions. The approach automatically detects moving ships in port surveillance videos with robustness for occlusions. *This work was published in the Journal of Electronic Imaging (2013) [108], in Emerging Research on Networked Multimedia Communication Systems (2015) [6], in Netherlands Conference on Computer Vision [124] and also in [125].*

The second use case focuses on crowd monitoring which is important for public street safety. This use case was part of the European iTEA ViCoMo (Visual Context Modeling) project, where contextual information was used to improve scene understanding. *The work presented in this use case was published in the Journal of Electronic Imaging (2013) [126] and also in Emerging Research on Networked Multimedia Communication Systems (2015) [6].*

The third use case addresses the use of contextual information to improve the detection of moving cars which are important objects in public road surveillance scenes. *This work was published in the IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)(2014) [91] and in the Proc. of the 4th Joint WIC/IEEE Symp. on Inform. Theory and Signal Proc. in the Benelux (2014) [4].*

The last use case addresses the detection of alarming situations both in private and public environments by analyzing the pixel-based motion context, as an alternative for the conventional object-based approach. *This work was published in the Proc. of the Int. Conf. on Image Proc., Computer Vision, and Pattern Recognition (IPCV)(2012) [127].*

At the end of this chapter, we will compare our gravity-based region labeling approach with a recently developed deep learning-based system in order to provide insight into how our approach performs compared to emerging deep learning technology. This chapter will present the details of the surveillance use cases in the order as indicated above.

6.2 Use case 1: Motion context and region labeling for moving ship detection in port surveillance

In port areas, various hazardous scenarios may occur caused by heavy traffic conditions and the mixing of large sea ships with local smaller vessels. In particular, dangerous situations can happen when small ships travel in the radar "shadow" of large ships, so that they become invisible for the radar system and the harbor management. Evidently, supplementary visual surveillance is a possibility but because of the large diversity of ship functionalities and shapes, human visual inspection is highly laborious and error-prone. Automatic ship detection is an attractive research topic in the field of port surveillance, which can nurture various applications, such as vessel traffic monitoring, ship identity management and smuggling prevention. We propose an approach to detect moving ships by jointly using the spatial (position) context, i.e. our semantic region labeling approach introduced in Chapter 4 and motion context [108]. This approach is a cooperative work of our semantic region labeling and the work of Bao *et al.* [108]. We now briefly explain our ship detection approach.

The approach is based on the following two observations in the scenario of port surveillance: (1) ships can only travel within the water region, (2) each ship has a spesific motion, which distinguishes itself from other ships and from the surroundings. Concerning the motion characteristics of ships, we note that each ship has its own motion pattern and its motion is more significant than the motion within the local background. The flowchart of our moving ship detection involves a sequence of processing steps, as depicted in Figure 6.1.

Let us briefly describe each processing step, as depicted in Figure 6.1. First, a graph-based segmentation [111] is employed to divide a video frame

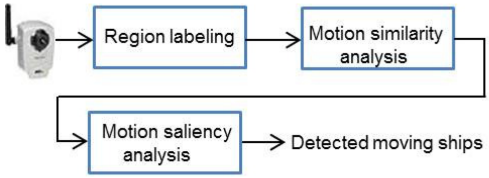


Figure 6.1: Schematic representation of context-based surveillance image and video analysis for detecting moving ships.

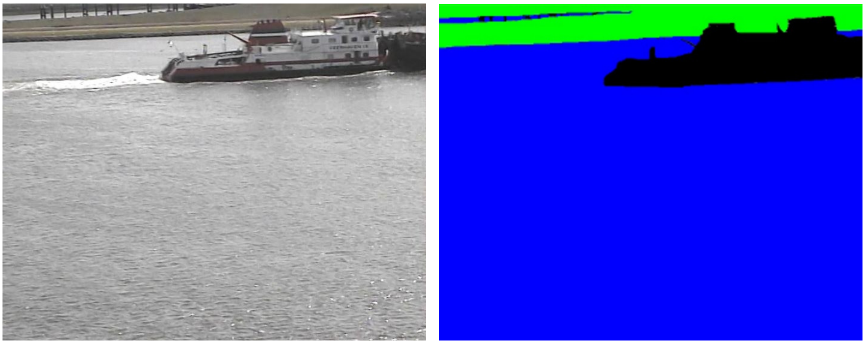
into segments. Then, our semantic region labeling approach introduced in Chapter 4 is employed to classify those segments into three classes: water, vegetation and "unknown". The labeled segments are then used to analyze the motion similarity. Adjacent segments with the same labels and statistically similar motion are merged into semantic regions, through which occluded regions are also separated from each other. These regions are analyzed based on semantic, spatial and scale constraints to provide knowledge of locations of candidate ships. Based on the common understanding that ships should have significant motion, the regions with salient motion are detected as moving ships. In this scenario, salient motion is defined based on a set of criteria to distinguish it from other types of motion, such as the scintillation/ripples of the water surface and the wind-based motion of vegetation. For more details about the proposed motion saliency we refer to the work of Bao *et al*. [108]. We start with the performance evaluation of the ship detection approach.

To evaluate the performance of the proposed ship detection approach, the algorithm is tested on real-life video sequences recorded in the harbor of Rotterdam, the Netherlands. All the videos were captured with a PTZ (Pan-Tilt-Zoom) camera with an SD resolution of 720×576 pixels. In those videos, the ships are of various types, including container ships, speed boats, tanker ships, fishing boats and sailing boats. All videos were made in the framework of the WaterVisie project, which will be called "WaterVisie dataset" in this section.

For the experiments, we employ 16 different video sequences. These sequences contain a significant amount of visual variation and are categorized into three scenarios: (S1) single/multiple ship without occlusion; (S2) ships present with occlusions between different ships and/or clutter caused by vegetation; (S3) ships during sunrise or sunset moment (highly flickering water).

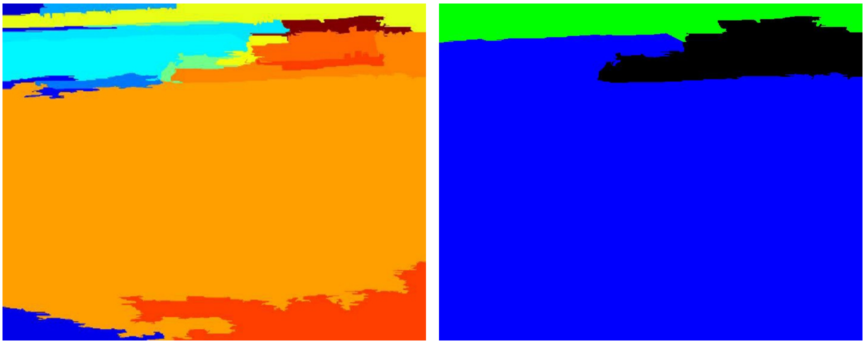
Figure 6.2 visualizes the result of our gravity-based region labeling approach on frames from WaterVisie video sequences. This dataset includes 3 different categories such as water, vegetation and possible ships which is labeled as "unknown" in our approach. As depicted earlier in this section, these semantically labeled regions as well as segmented regions are exploited as contextual information. Despite the lack of color information, it is obvious that the gravity model performs promising.

Since the ship detection system is based on a pan-tilt-zoom (PTZ) camera, it is hard to benchmark this system, because existing systems are mainly



(a) Sample frame of WaterVisie video sequence

(b) Manually annotated regions



(c) Result of graph-based segmentation

(d) Our gravity-based labeling results

Figure 6.2: Region labeling results for WaterVisie dataset. From left column to right column: Frames from WaterVisie dataset; Ground-truth of corresponding frames; Intermediate results of graph-based segmentation. The colors are randomly chosen and are not related to semantic class labels; Region labeling results using gravity model (Green=Vegetation, Blue=Water and Black=Unknown).

based on a static camera. Furthermore, there is no benchmark dataset to evaluate ship detection systems in general. Therefore, the performances of different ship detection techniques are difficult to compare. As such, to analyze the proposed approach, we compare it with the "Existing" [12] and "Cabin detector" [128] approaches.

Figure 6.3 presents a visual comparison among "Cabin detector", "Existing" method and the proposed ship detection. The "Cabin detector" proposed by Wijnhoven *et al.* [128] utilizes local descriptors for representative parts of ships instead of modeling the complete ship appearances. They build a cabin detector based on HOG (Histogram of Oriented Gradients) [129] and classify the resulting patterns. However, the simplified local descriptors are hardly distinctive from other highly textured patches in the image, such as vegetation. Moreover, the algorithm fails on ships without cabins. "Existing" method is the algorithm that exploits only water regions as contextual information and then applies the motion saliency detection technique. The "Existing" approach is a basis of the proposed ship detection algorithm which will be called the "Improved" approach in this section. The "Improved" approach uses not only water regions, but also vegetation area as contextual information and then applies the motion saliency detection. Therefore, it can handle occlusions between ships as well as clutters between ships and vegetation. The first row shows the results for the "Improved" ship detection approach, the second and third rows are the corresponding results of the "Existing" and "Cabin detector" approaches, respectively. The four presented frames demonstrate the categorized scenarios from left to right: (S1) a long vessel without occlusion; (S2) a ship cluttered by vegetation; (S2) two ships occluding each other; (S3) a sailing ship during sunrise moment.

For all scenarios, the "Improved" approach can successfully find the whole ship with a bounding box indicating the delineation of the ship's body. In contrast, the "Cabin detector" can only mark the cabin parts of the ship (Figure 6.3 (j)) or it generates several detections along the ship body (Figure 6.3 (i)). For small ships moving in the flickering water region, the "Cabin detector" misses the target (Figure 6.3 (l)), while the "Improved" method can still find the ship with a boundary indication (Figure 6.3 (d)). Figure 6.3 (f) shows an example where a ship is cluttered by vegetation. In Figure 6.3 (g), the detection fails because the two ships traveling in opposite directions are regarded as one ship, which makes the motion of the object not salient compared to the surroundings.

Table 6.1 shows a quantitative evaluation of the detection results for the three ship detection algorithms. In this evaluation, only the "miss or hit"

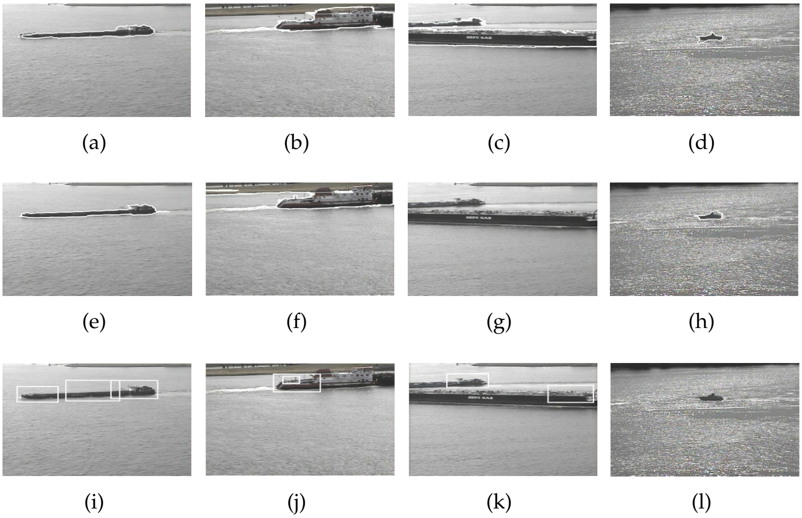

(a) (b) (c) (d)
(e) (f) (g) (h)
(i) (j) (k) (l)

Figure 6.3: Visual comparison among our "Improved" ship detection approach, Existing method and Cabin detector. The first row shows the results for our Improved ship detection approach; the second and third rows are the corresponding results of the Existing method and Cabin detector. The 4 typical frames demonstrate the categorized 3 scenarios from left to right: a long vessel without occlusion(S1); a ship cluttered by vegetation(S2); two ships occlude each other(S2); a sailing ship during sunrise moment(S3).

is considered, which means that the detection is successful even if the detected ship contains a certain portion of non-ship objects. In Scenario 1, the proposed ship detection approach using context information successfully detects 1,413 ships out of 1,593 ships, with a total precision of 94.5% and a recall of 88.7%. It gains approximately 2% in both precision and recall compared to the "Existing" method, directly benefiting from the more advanced context model. The "Cabin detector" system obtains similar recall (87.2%) at the cost of a low precision (65.1%). This is caused by the fact that the developed appearance model in the "Cabin detector" approach is simplified, but not distinctive enough for other non-ship textured objects in the scene. Therefore, it tends to generate false detections in vegetation or redundant detections along long ships. In Scenario 2, the "Improved method" provides significantly higher recall (92.7%) compared to the "Existing" (70.3%) and "Cabin detector" (41.8%) approaches. The precision of the three approaches is similar in this scenario. In Scenario 3, the "Improved method" provides higher precision (96.3%) and recall (75%) compared to “Existing” (precision of 93.8% and recall of 71.8%) and ‘Cabin detector” (precision of 84.3% and recall of 56.1%) methods.

In summary, the above results show that the “Improved method” outperforms the “Cabin detector” when a flickering background affects the ship appearance severely (e.g. sunrise in Figure 6.3 (b) and Figure 6.3 (j)). Since the “Improved method” avoids using the detector which is trained for finding ship appearances (“Cabin detector”), it still performs well when the target ship differs from the training samples. However, the “Cabin detector” relies on frame-based features, so that the performance significantly deteriorates, which is caused by the water flickering. Comparing between the “Improved method” and the “Existing” method, the higher values in both precision and recall obtained by the “Improved method” demonstrate the advantage of the fully-modeled context-based approach.

Test Videos	Methods	TP+FN	TP+FP	TP	Precision (%)	Recall (%)
S1	a. Improved method	1593	1496	1413	94.5	88.7
	b. Existing	1593	1491	1374	92.1	86.3
	c. Cabin detector	1593	2135	1389	65.1	87.2
S2	a. Improved method	455	433	422	97.5	92.7
	b. Existing	455	325	320	98.5	70.3
	c. Cabin detector	455	207	190	91.8	41.8
S3	a. Improved method	173	135	130	96.3	75.0
	b. Existing	173	130	122	93.8	71.8
	c. Cabin detector	173	115	97	84.3	56.1

Table 6.1: Ship detection results. $TP + FN$ = manually marked ships, $TP + FP$ = detected ships, TP = correctly detected ships.

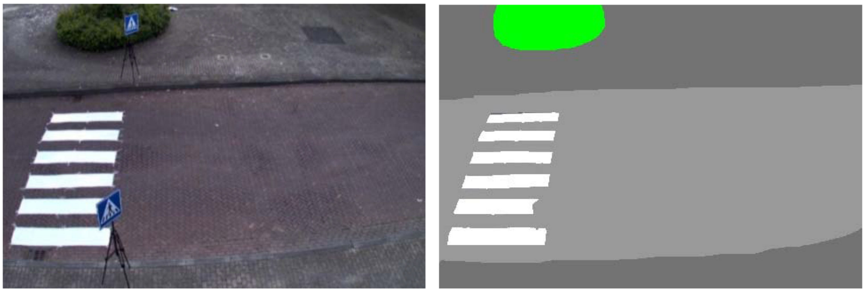
The next section shows how reliable contextual aspects, such as semantic labeled regions and traffic sign contexts, lead to semantic understanding of a scene.

6.3 Use case 2: Automatic traffic sign detection and region labeling for traffic action recognition

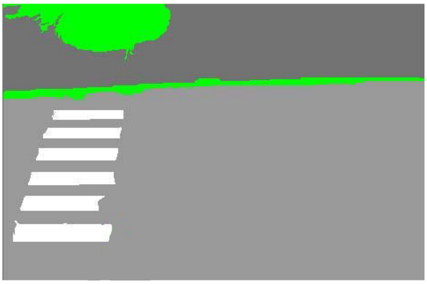
In traffic surveillance video scenes, not only harbor monitoring, but also street traffic is of high importance. This involves the monitoring of large groups and crowds of people and associated behavioral analysis. Many algorithms exist to detect and track single persons, but analyzing group behavior is a relatively unexplored area of research. The work presented in this section is part the European iTEA ViCoMo (Visual Context Modeling) project. One of the central goals of the ViCoMo project was using contextual information to improve scene understanding, and this section describes the result from the collaboration between several ViCoMo partners.

Figure 6.6 illustrates the result of our gravity model for our surveillance application. This dataset includes four different categories such as road, vegetation, construction and zebra crossing regions. As indicated earlier in this section, these semantically labeled regions are exploited as contextual information.

We propose a system for automatically analyzing group behavior using contextual clues to interpret and classify the actions of the group of people. Our hypothesis is that traffic signs together with semantically labeled regions



(a) Sample frame of our video sequence (b) Manually annotated semantic regions



(c) Our gravity-based labeling results

Figure 6.4: Our gravity-based region labeling results on the video sequence (Green = Vegetation, white = zebra crossing, light gray = road and dark gray = construction).

(e.g. zebra crossing in this case) provide contextual information for surveillance applications and help to interpret the actions of traffic participants, as well as to detect illegal or dangerous traffic situations automatically. Evidently, this list of contextual clues, i.e. traffic signs together with semantic labeled regions, is not exhaustive, but for this system we limit ourselves to these aspects.

Our system contains three components: (1) the group analysis algorithm [126], to track the movement and any splits/merges of groups of people, (2) our semantic region labeling approach introduced in Chapter 4, and (3) the traffic sign detector [3]. Finally, an event-level decision engine interprets the output of these algorithms and decides when an alarm needs to be initiated, see Figure 6.5 for an overview of the system. We now briefly describe each processing step, as depicted in Figure 6.5.

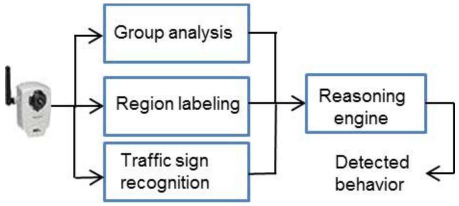


Figure 6.5: System overview of the group behavior analysis system.

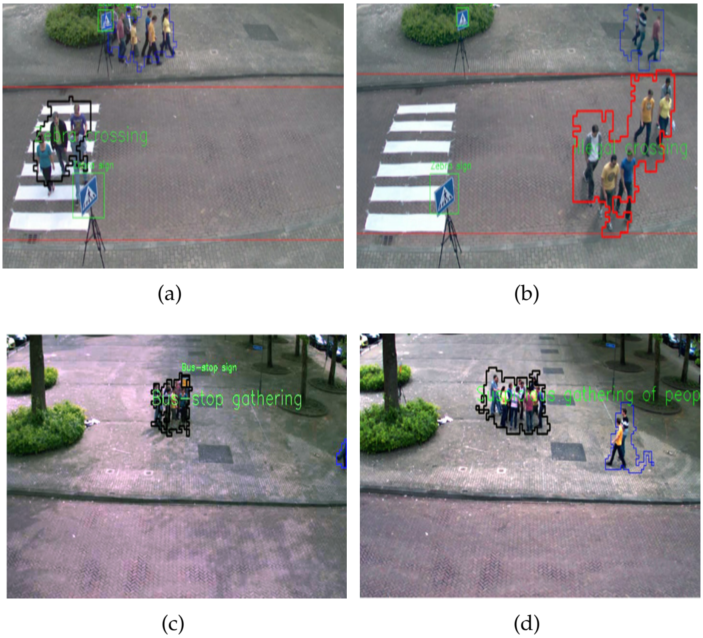
The group analysis algorithm [126] rests on a motion-based, rather than object detection-based framework that localizes groups. The group analysis approach also has the ability to detect pixel/motion-based group events, such as merges and splits of groups of people. The motion estimation algorithm is based on the Lucas–Kanade optical flow algorithm. In order to classify group events, the consistency of behaviour is assessed by finding spatial blob matches between frames and checking whether the matching record is continuous over a substantial number of frame intervals. For more details about the group analysis algorithm, the reader is referred to [126].

The proposed traffic sign detector is summarized as follows. First, traffic signs are detected in the images using independent detectors for all sign appearance classes (i.e. red circular signs), using a multi-scale sliding window approach. This leads to a detection bounding box and a sign class. Next, the detected signs are classified to determine the exact type of traffic sign, using the extracted bounding boxes. The detection algorithm locates traffic signs in the images. Multiple detectors are used to find broad classes of traffic signs.

For example, all red circular signs are detected using a single detector, but a different detector is used for red triangular signs. For more details, the reader is referred to [3].

The system is tested with the recorded video in an outdoor setting, containing several acted scenarios. We present an example case to demonstrate a situation where traffic signs help in the understanding of surveillance footage. In Figure 6.6, the security camera footage shows several scenes of actors in two scenarios, and the actors and traffic signs in the scene are automatically detected. The detected actors and signs are analyzed by a decision engine, which decides if the situation requires operator's attention. In the first scenario (Figure 6.6 (a) and 6.6 (b)), a zebra crossing and a traffic sign are first detected. This detection provides information to identify whether people are crossing the road in an illegal location (Figure 6.6 (b)), which may lead to a dangerous traffic situation. In the second scenario, two contrasting scenes are shown (Figure 6.6 (c) and (d)). First, a crowd gathers near a bus stop and waits for a bus to arrive, which is a perfectly normal situation and can be identified as such through the bus-stop sign (Figure 6.6 (c)). In the second scene, a crowd gathers to watch a fight, which is not a normal situation and requires security operator attention (Figure 6.6 (d)). Screenshots from the first two scenarios are visualized in Figure 6.6. In order to enable a fair comparison of our work to already published research, we have evaluated the performance of our system using publicly available datasets. Since these datasets are not fully suitable to test our system, we can only report the performance of a subset of the complete functionality of our system. We focus on motion-based group event detection, tracking groups and detecting splits, merges and slide-by (occluding groups passing each other without merging) events. For this evaluation, we have used the publicly available PETS 2004 and 2009 datasets. Because these datasets only contain a limited amount of relevant group events, we have opted to add some of our own test sequences to the dataset, to increase the statistical significance of the comparison experiment.

We report two precision-recall scores, one for the detection of basic crowd behavior events and one for the detection combined with a correct classification of the event type (split, merge or slide-by). We consider repeated detections of the same event to be false positives. The dataset contains 89 events with 33 merges, 31 splits and 25 slide-by events, from 33 video sequences. The system exhibits a detection recall of 83% with a corresponding precision of 37%. The scores for combined detection and subsequent correct classification are 71% and 32% for recall and precision, respectively. The low value



(a) (b) (c) (d)

Figure 6.6: Screenshots from the zebra-crossing scenario ((a) and (b)), and the bus-stop/fight scenario ((c) and (d)).

for precision can be largely explained by the repeated detections, shadows, motion of limbs within the group and stationary objects that appear in the foreground (between group and camera).

In literature, there are a few systems that also report performance figures on PETS data sets. Garate *et al*. [130] have tested their crowd event recognition system on the PETS 2009 data set, and have reported a recall of 79% for splitting, and 60% for merging events. Chan *et al*. [131] have reported error rates of 23% and 32% for classifying splitting and merging events, respectively. These scores are all relatively close to our results, indicating that our performance is comparable to the state-of-the-art systems discussed in recent literature.

It should be noted that the traffic action-recognition scenarios are complex. As such, the dataset had to be partially captured by a group of volunteers and the number of experiments at the scenario level is limited. Given this constraint, it has been found that the proposed traffic action-recognition system evaluations prove to be well working for the dataset at hand. This positive result is due to the considerable testing of the individual components and the relative simplicity of the decision engines for the scenario content.

In traffic surveillance video scenes, a moving object such as a car should be detected to provide semantic information and semantic image regions of interest are road and vegetation. The next section shows how reliable contextual aspects, such as automatically labeled regions and detected moving cars, lead to semantic understanding of a scene.

6.4 Use case 3: Detection of moving cars in traffic surveillance

In this section, two systems are introduced which refer to automatic moving car detection in traffic surveillance videos. In the first system, our semantic region labeling algorithm presented in Chapter 4 is combined with motion information. In the second system, our DCT-based salient region detection technique introduced in Chapters 3 is combined with a car detection approach.

6.4.1 Motion context and region labeling for car detection in traffic surveillance

In video-based traffic surveillance systems, the detection of moving objects of interest is an important research topic in computer vision for traffic surveillance. This importance is motivated by that the properties, the number and locations of moving objects are fundamental to semantic interpretation of traffic information. Such objects of interest can be defined depending on the scenario of surveillance, e.g. vehicles in road surveillance or vessels in port surveillance. In the past decades, significant research concentrated on object detection in the domain of traffic surveillance. The object detection algorithms discussed in this research are mostly designed and implemented in a dedicated form for a particular traffic scenario: for example, they focus either on road surveillance or on port surveillance. Therefore, these dedicated algorithms lack generic characteristics, which hampers their re-use for object of interest detection, independently of the domain. In this section, we propose a context-based generic framework combining scene understanding and detection of objects of interest for a traffic surveillance system. The framework is inspired by the improved ship detection algorithm presented in Section 6.2 and conceptually further developed for completeness and generalization. Figure 6.7 illustrates the proposed framework. The proposed framework consists of five stages as depicted in Figure 6.7, which are discussed here.

First, our semantic region labeling algorithm proposed in Chapter 4 is performed to divide the video frame into labeled segments, such as road, vegetation, etc. Those segments are then grouped into regions according to spatial and motion similarities. Therefore, the scene is depicted as a set of semantic regions and the first group of candidate objects is located simultaneously.

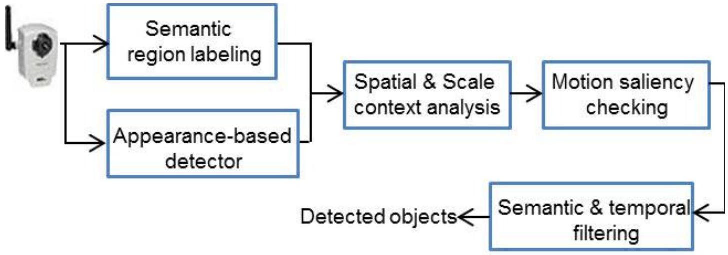

Semantic region labeling
Appearance-based detector
Spatial & Scale context analysis
Motion saliency checking
Semantic & temporal filtering
Detected objects

Figure 6.7: Schematic representation of context-based surveillance image and video analysis for detecting cars.

Meanwhile, as a second stage, a simple appearance-based detector is trained off-line and applied to the same frame to locate the second group of regions possibly containing objects of interest. Haar-like features are used in our framework because they are more easily computed than HOG features. The boosted classifier is used for training the detector, considering this weak classifier is faster, compared to SVM.

The third stage is extracting spatial context. In a specific traffic surveillance scene, an object of interest is supposed to travel inside the traffic region, e.g. vehicles travel in/on a road region and ships travel inside the water region. Therefore, we assume that regions containing candidate objects should be surrounded by traffic regions or at least have common borders with them.

In the fourth stage, we verify whether the motion of the obtained candidates is salient compared to their local surroundings. Using the pixel-based motion obtained by optical flow, we calculate the region-level motion, based on the average motion in each region. Then, we remove/filter false positives whose motion contrast with the surroundings is not significant (e.g. traffic signs). In the second criterion, we remove false detections (e.g. swaying flags) with small distracting motion in a static background region.

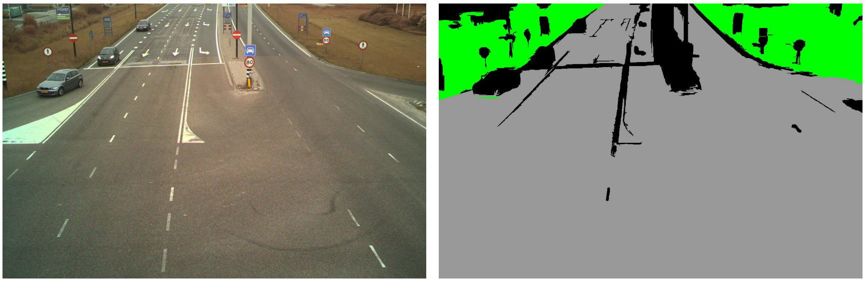
Finally, the detection results are merged and a temporal filter is applied to increase the robustness of the detection results. The temporal filtering is based on the assumption that a moving object cannot disappear suddenly from the scene. For each detection in the previous frame, we search a predefined neighboring area for detections in the current frame. If no detection is found in the search area, the previous detection is propagated to the current frame.

To validate our context-based traffic surveillance analysis framework, we consider a road traffic scene and on-road vehicles as an example. All videos have a resolution of 1280×960 pixels and are captured during daytime, including sunny and cloudy weather. For training the vehicle detector, we choose 100 images from the "cars 2001(Rear), Caltech" dataset. The training set is intentionally chosen to be limited to a single view with small number of training images, in order to evaluate the system performance when an imperfect vehicle detector is applied. The semantic region labeling from Chapter 4 aims at annotating 2 classes (vegetation, road) plus the class "unknown". Figure 6.8 visually demonstrates a sample input frame (Figure 6.8(a)), semantic labeled regions (Figure 6.8(b)) and the output of our vehicle detector.

To evaluate the performance of our vehicle detector, we compare our approach with another algorithm proposed by Wijnhoven *et al.* [132]. The approach of Wijnhoven *et al.* [132] is based on the HOG (Histogram of Oriented Gradients) and the SVM classifier. Figure 6.9 provides a visual comparison of our approach and the approach of Wijnhoven *et al.* [132]. In this figure, the images at the left column show the results of our system, and the images at the right column illustrate the results of Wijnhoven's approach. In contrast to our approach, the Wijnhoven's work erroneously detects several traffic signs as vehicles (Figure 6.9(a)). Furthermore, the turning cars (Figures 6.9(b)) and occluded cars (Figure 6.9(c)) are missed in the Wijnhoven's results. In contrast, our approach shows successful detections in all three scenarios. In the next section, we cascade our DCT-based salient region detection technique from Chapters 3 with the Haar-based car detector presented in this section to measure the potential efficiency gain.

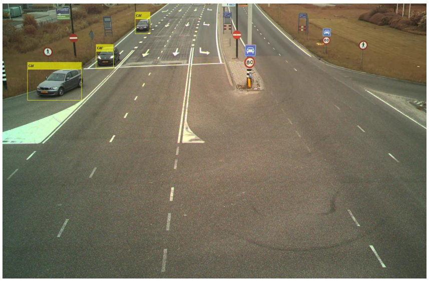
6.5 Motion context and salient regions for detecting moving cars in traffic surveillance

To measure the potential efficiency gain when applying salient region detection in video surveillance applications, we cascade our DCT-based salient region detection technique from Chapters 3 with an object detector [133], as depicted in Figure 6.10. Here, the cascaded detection framework is again



(a) Sample frame of our video sequence

(b) Semantic labeled regions



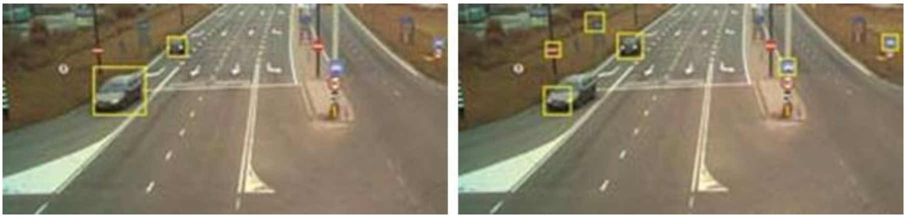
(c) Car detection results indicated by yellow bounding boxes

Figure 6.8: Our proposed car detector.

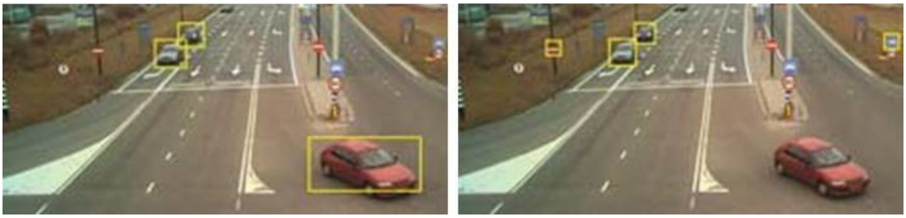
evaluated for car detection task and is inspired by the car detection algorithm presented in Section 6.4.1 and conceptually updated for achieving the potential efficiency gain.

The proposed framework consists of two stages as depicted in Figure 6.10, which are discussed here. First, we apply our DCT-based salient region detection technique introduced in Chapters 3 to provide informative regions for our car detector. Then, we apply our car detector on each salient region, instead of analyzing the complete scene. Our car detector is the same as presented in Section 6.4.1 which is trained using Haar-like features combined with Adaboost algorithms.

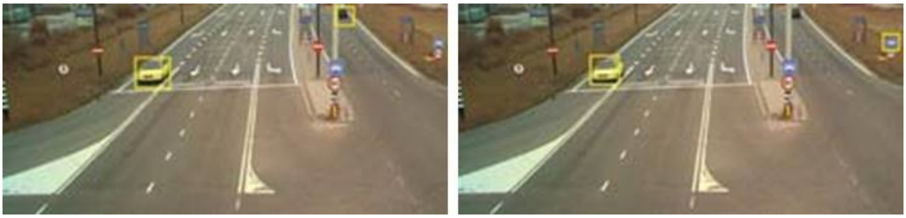
To train the car detector, we choose 100 images from the “cars 2001(Rear), Caltech” dataset [134] as training set. To test our detection framework, we choose 50 frames from our highway video sequence which was introduced in Section 6.4.1.



(a) Multiple vehicles approaching the crossing



(b) Vehicle is turning (views are changing)



(c) Vehicle is occluded by light

Figure 6.9: Detection results of two approaches in different scenarios, where left column shows results of our "Context-Generic" and right column shows "SVM-HOG".

We consider the size of the car in the test set to be in the range of 20×20 pixels and 120×120 pixels. We use a sliding window with the size of 120×120 pixels to scan the image. The scanning step is 10 pixels in both vertical and horizontal directions. If the number of pixels in the sliding window that are also within the detected salient region is below $50\ \%$, we expect no car inside the image patch and therefore we do not apply the car detector. Otherwise, we apply the car detector for the image patch.

Figure 6.11(b) shows the result of the DCT-based salient region detection approach with 8×8 block size on one sample frame of our highway video sequence (6.11(a)). Figure 6.11(c) presents the detected cars in that frame us-

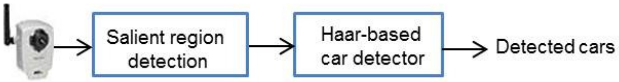
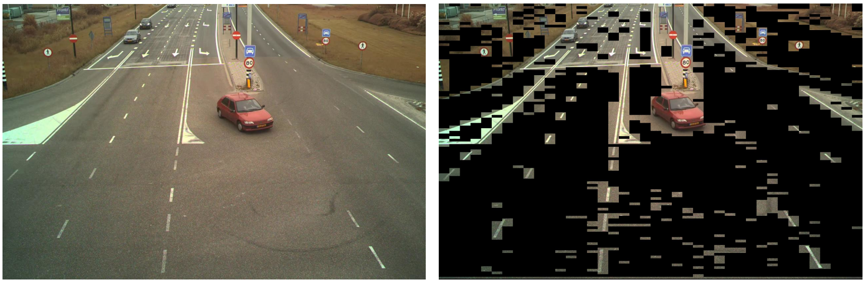


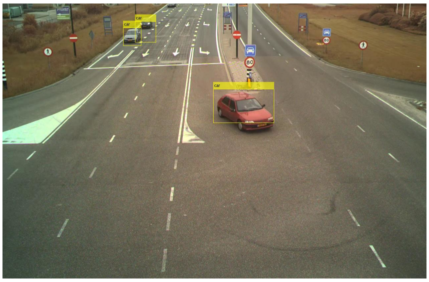
Figure 6.10: Framework of the cascaded object detection approach.

ing the DCT-based salient region detector.

To evaluate our proposed approach, we compare the detection results of our cascaded framework with the results obtained using only the Haar-like object detector. On average, for 50 frames of our test set without using the detected salient regions, the car detector is applied to 9,744 image patches; while applying the detector only on salient regions, only 943 image patches are checked by the car detector. Figure 6.12 illustrates the visual comparison



(a) Sample frame of our video sequence (b) DCT-based salient region selection

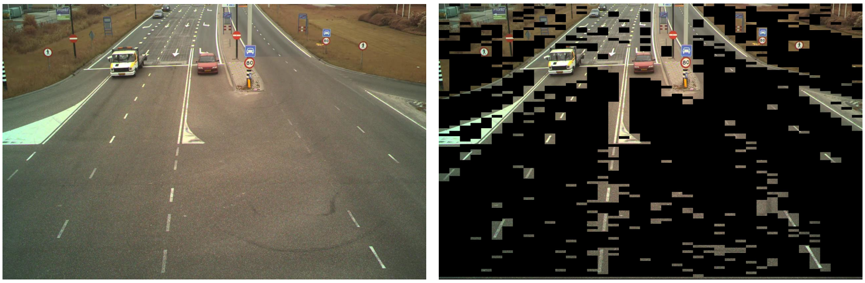


(c) Our context-based car detector using salient region detection

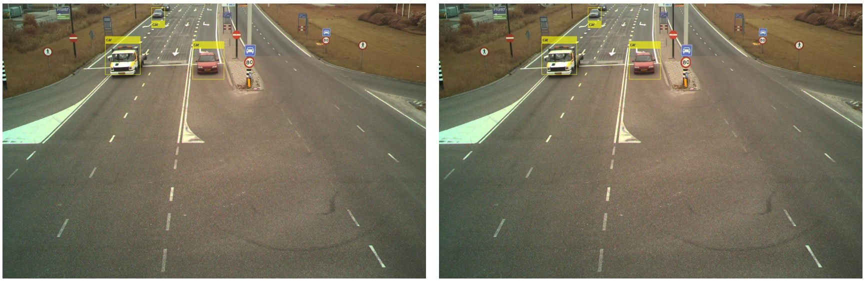
Figure 6.11: Proposed cascaded car detector (yellow bounding boxes represent the results of the car detector).

of a Haar-like car detector approach with (Figure 6.12(c)) and without (Figure 6.12(d)) DCT-based salient region detector (Figure 6.12(b)) on one sample frame of our highway video sequence ((Figure 6.12(a)). The proposed cascaded detection framework using DCT-based salient region detection technique leads to applying the car detector, which is a computationally expensive algorithm, to only 9.7 % of the total amount of image pixels. The average detection rates of car detection with and without using salient regions are 80 % and 90 %, respectively.

In the surveillance domain one of the interesting challenges is the detection of abnormal-events. In the next section, our research is presented on exploiting motion information in a fast abnormal event detection system.



(a) Sample frame of our video sequence (b) DCT-based salient region selection



(c) Our context-based car detector without using salient region detection (d) Haar-based car detector using salient region detection

Figure 6.12: Our cascaded car detector.

6.5.1 Use case 4: Fast abnormal-event detection from video surveillance

This section presents a novel block-based approach to detect abnormal situations by analyzing the pixel-based motion context, as an alternative for the conventional object-based approach. We proceed directly with event characterization at the pixel level, based on motion estimation techniques. Optical flow is used to extract information such as density and velocity of the motion. The proposed approach identifies abnormal motion variations in regions of motion activity, based on the entropy of DCT coefficients. We aim at a simple block-based approach to support a real-time implementation. In this use case successful results are reported on the detection of abnormal events in surveillance videos captured at an airport. We now briefly describe our approach.

Our abnormal event detection is based on motion features extracted with a motion estimation technique. We base our features on pixel-based optical flow, since this is the most natural technique for capturing motion independent of appearance [127]. We use optical flow at each frame using the Lucas-Kanade algorithm [115]. Optical-flow-based motion estimation uses characteristics of flow vectors of moving objects over time to detect moving regions in an image sequence, relating each image to the next. Each vector represents the apparent displacement of each pixel from frame to frame [49]. The result of optical flow is the value of the displacement of each pixel in both vertical and horizontal direction. We combine this displacement to obtain a motion magnitude vector. To process these motion vectors, we substitute pixel values for the estimated motion (called motion image) and we divide each motion image into blocks.

We expect that during abnormal events the motion patterns and therefore the energy of the images containing motion vectors, change compared to the normal behavior. We apply a DCT to each block of the motion image, since it provides a better representation of the motion pattern. Then we compute the entropy of the DCT coefficients to measure the information content of the DCT coefficients [135]. The details of the entropy calculation is described in Chapter 2. In our case, the motion magnitude per pixel forms a gray-scale image and we use 256 bins which correspond to the number of gray levels.

For deciding whether the event is normal or abnormal, we compare the entropy value with thresholds which we learn per block in the beginning of the video sequence. The threshold is based on a median value of the entropies which we estimate during the first 500 frames of the video. Obviously, we assume that abnormalities will not occur during the first 500 frames of the surveillance sequence. In general, this median computation of the threshold can be done continuously through the video because the abnormalities will be filtered out by taking a median value. We limit the median filtering over time for controlling the total complexity of the calculations. An abnormal event is indicated when the value of the entropy for the current block is higher than the threshold defined for that block. In our implementation, we divide each frame into 4 blocks for simplicity, but this can be easily changed. Based on experimentations and evaluations, the threshold for the median entropy to classify an abnormal event is empirically set to 3 times the median value. The abnormality indicator for the whole frame is raised if abnormality is detected in any of the blocks. Figure 6.13 illustrates our block-based processing framework in dynamic scenes.

Figure 6.14 illustrates the result of our system in a normal situation where people are waiting in a queue. Figure 6.14 (a) shows a frame of the normal situation and in Figure 6.14 (b) we indicate the presence of the alarm in an analysis window. Since the situation is a normal behaviour in the airport, the analysis window does not show any alarm.

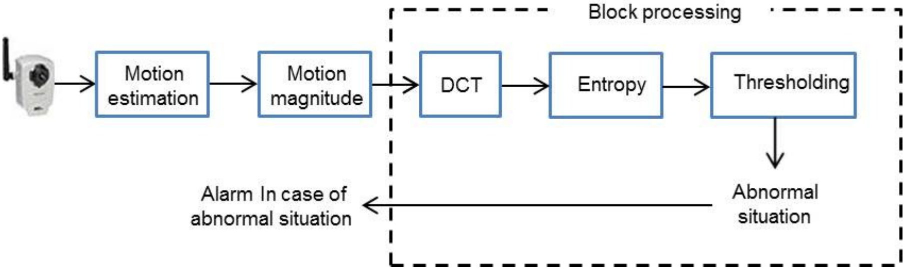
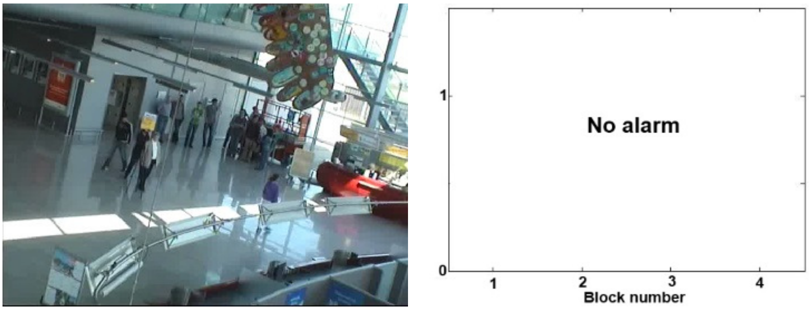


Figure 6.13: Block-based processing framework for detecting an abnormality.

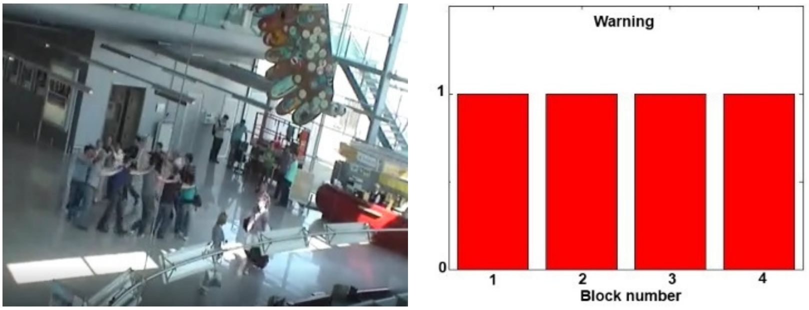
Figure 6.15 illustrates the result of our system in an abnormal situation where people are moving in a circle. The original frame of the airport video sequence (6.15 (a)) shows the abnormal situation where people are moving in a circle and the analysis window (Figure 6.15 (b)) shows a clear warning with the 4 bars on all 4 blocks. From this video sequence, our approach detected 5 out of 6 abnormal events, without producing false alarms.

For comparison, we use an alternative approach proposed by Ihaddadene and Djeraba [106]. In this approach, a statistic measure is defined which describes how much the optical flow vectors are organized or cluttered in each frame. This alternative algorithm uses a metric which is the scalar product of the normalized values of the following factors: direction variance, motion



(a) (b)

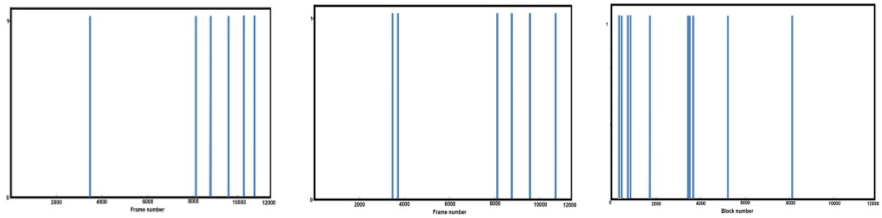
Figure 6.14: Result of our system in a normal situation.



(a) (b)

Figure 6.15: Original frame with a result of our system in an abnormal situation. Each bar indicates an alarming situation in the corresponding block (block numbering is column-wise, starting at the top left).

magnitude variance and direction histogram peaks. This metric is then compared to a threshold which is manually set at a constant value. We have applied the approach of Ihaddadene and Djeraba [106] to our video sequences. It is noted that in airports, people normally move in any desired direction, therefore we do not consider their movement direction as a suitable feature which has been used in [106]. After experimenting with the original approach of these authors [106], we have tried to improve their results by considering only the motion variance per frame. We have experimented with several threshold values for this approach and the best obtained results are illustrated in Figure 6.16 (c). We can observe that this method produces false alarms in normal situations and it has missed 3 abnormal events.



(a) (b) (c)

Figure 6.16: (a) Ground-truth results in block number 3 for the complete surveillance video, (b) warnings based on our detection corresponding to those events (bars show alarm in block number 3 at the current frame and the first two bars correspond to one event) and (c) warning for comparison corresponding to the system of [106].

6.6 Comparison between our gravity-based and a deep learning-based region labeling system

In this section, we compare our gravity-based region labeling with a recently developed region labeling system [2] which is based on emerging deep learning technology. We apply both our region labeling approach and the deep learning-based approach to a new dataset which is called Cityscapes [2]. Furthermore, the deep learning-based approach of Meletis *et al.* is applied to the LableMe dataset that we used in our previous chapters. Prior to presenting present the results, first we provide a brief description of the deep learning approach.

The proposed region labeling system by Meletis *et al.* [2] first selects a dataset (primary dataset) with fine (sufficiently annotated), per-pixel annotations, that contains the high-level classes, e.g. vehicles, humans, traffic signs. Then, a collection of datasets (auxiliary datasets) is selected, with no limitations on annotation type, that contain lower level (sub)classes, e.g. vehicle or traffic sign types, of the primary or another auxiliary dataset. The next step is to construct the semantic tree of the label spaces from the selected datasets. At the tree root, the label space of the primary dataset must be placed. The label spaces of the auxiliary datasets are placed one-by-one below their corresponding higher-level class of an already added dataset. Figure 6.17, depicts an example of a two-level hierarchy including 5 datasets with various annotation types. The hierarchy of the datasets label spaces induces the corresponding hierarchy of classifiers. Each classifier is associated with its respective label space and the created tree of classifiers is trained, in an end-to-end, fully convolutional manner, over a shared feature representation.

The proposed network architecture consists of a fully convolutional feature extractor for computing a dense, shared representation, and the interconnected classifiers that correspond to every label space of the semantic hierarchy, as shown in Figure 6.18. Each classifier is preceded by a shallow subnetwork, which adapts the common representation, its depth, and receptive field to the needs of the classifier and the corresponding dataset's character-

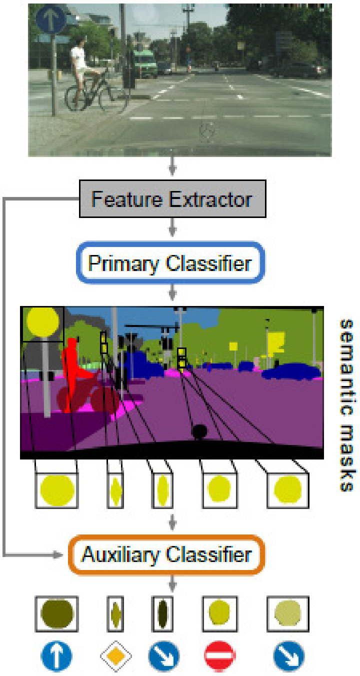


Figure 6.17: Application of Meletis *et al*. method [2] to a two-level hierarchy containing high-level street scene classes and traffic sign subclasses.

istics. For example, discriminating between e.g. traffic signs is easier [2], as less feature layers are needed, compared to high-level classes, like road vs. sidewalk and bushes vs. trees [2].

The feature extractor in Figure 6.18 consists of the feature layers of the

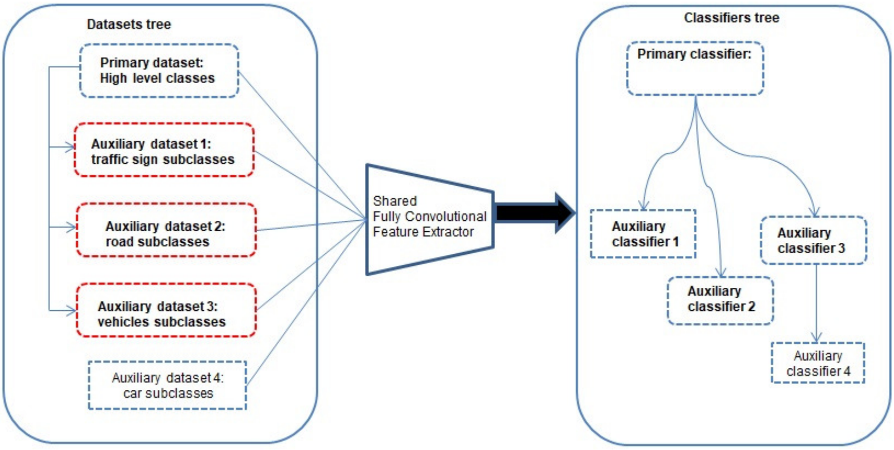

Datasets tree
Primary dataset: High level classes
Auxiliary dataset 1: traffic sign subclasses
Auxiliary dataset 2: road subclasses
Auxiliary dataset 3: vehicles subclasses
Auxiliary dataset 4: car subclasses
Shared Fully Convolutional Feature Extractor
Classifiers tree
Primary classifier:
Auxiliary classifier 1
Auxiliary classifier 2
Auxiliary classifier 3
Auxiliary classifier 4

Figure 6.18: Conceptual illustration of the method proposed by Meletis *et al*. that extends the number recognizable classes on the primary dataset using heterogeneous auxiliary datasets. The selected datasets are placed in a semantic hierarchy and the corresponding hierarchy of classifiers is constructed. Each classifier adapts a common fully convolutional feature representation and outputs per-pixel decisions, which are combined according to the hierarchy to provide the final result.

ResNet-50 architecture [2]. The stride on the input is reduced from 32 to 8, using dilated convolutions. The shared representation has a depth of 2,048, spatial dimensions 1/8 of the input, and is shared among two classifier branches. Then a 1x1 convolution layer (with ReLU and Batch Normalization) follows, to decrease feature dimensions to 256, a common technique also followed in [136] (150 and 100 feature dimensions, respectively). At the end of the two branches, two softmax classifiers are attached, each of which include a 1×1 convolutional layer (without nonlinearity) for producing the logits and the hybrid upsampling. The feature dimensions and the field-of-view of the per-classifier adaptation subnetworks are set to be the same for the two branches. Based on experiments of the number of layers for the best performance , a hybrid upsampling is proposed. This hybrid upsampling consists of one learnable fractional strided convolution layer (deconvolution), for doubling the resolution, followed by bilinear upsampling to reach input dimensions.

In this chapter our gravity-based and the deep learning-based approaches have been applied on two datasets: Cityscapes [2] and “LabelMe” [107] with the frame size of 256×256 pixels. Figure 6.19 and Figure 6.20 illustrate the results of our gravity-based and deep learning-based semantic region labeling approaches on the datasets. As it can be seen in Figure 6.19, the deep learning-based approach shows a slightly better result in detecting traffic signs as construction, compared to the results of our gravity-based approach. Figure 6.20 displays slightly better results for recognizing “sky” and “road” regions compared to the results of the deep learning-based approach.

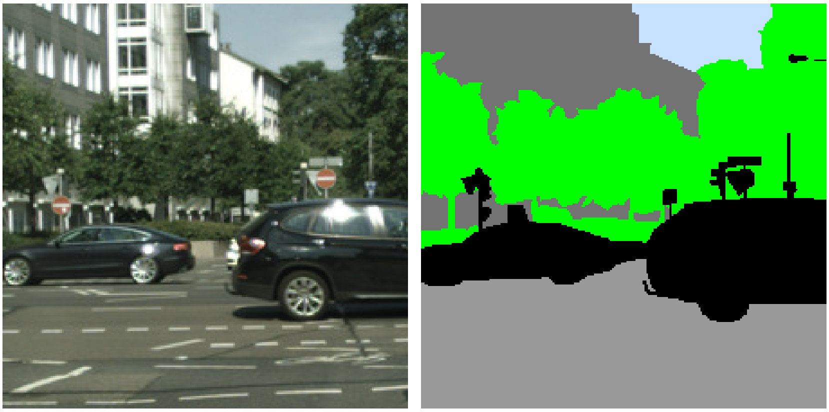
Table 6.2 demonstrates the accuracy of deep learning-based [2] and our gravity-based semantic region labeling approaches on the Cityscapes dataset over four region classes. Both approaches show comparable results. The deep learning-based approach performs better over vegetation, construction and road classes. Table 6.3 demonstrates the accuracy of deep learning-based [2] and our gravity-based semantic region labeling approaches on the Cityscapes dataset over four region classes. Both approaches show comparable results. Our gravity-based approach presents slightly better results compared to the deep learning-based method over road class and the deep learning-based approach performs slightly better compared to our gravity-based approach over construction class.

Region labeling approaches	Sky	Vegetation	Construction	Road
Deep learning-based	99	98	98	99
Our gravity-based	99	93	95	97

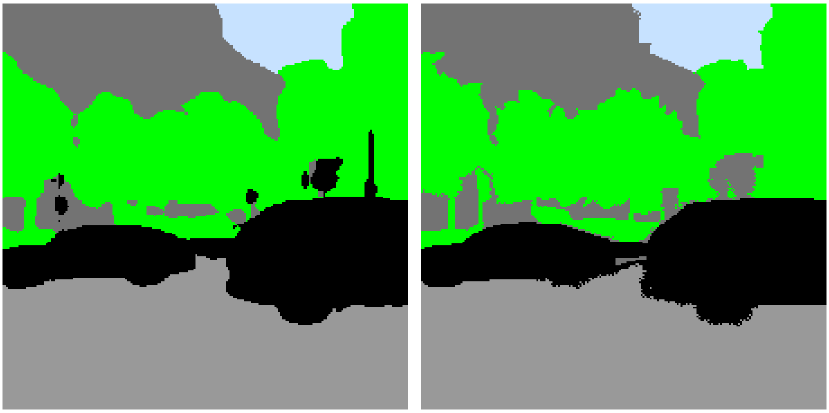
Table 6.2: Accuracy (%) comparison for the semantic labeling algorithms on Cityscape dataset (14 images).

Region labeling approaches	Sky	Vegetation	Construction	Road
Deep learning-based (Over 20 images)	98	97	95	97
Our gravity-based (Over 40 images)	98	97	93	98

Table 6.3: Accuracy (%) comparison for the semantic labeling algorithms on LabelMe dataset.



(a) Image from Cityscapes (b) Manually annotated regions

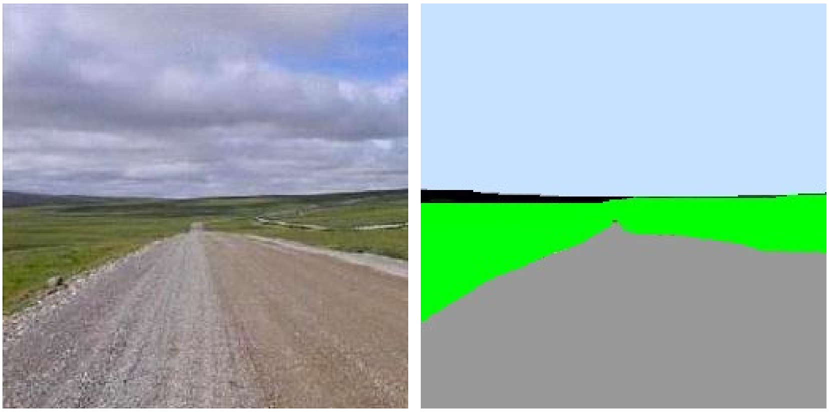


(c) Deep learning-based region labeling of (a) (d) Our gravity-based region labeling of (a)

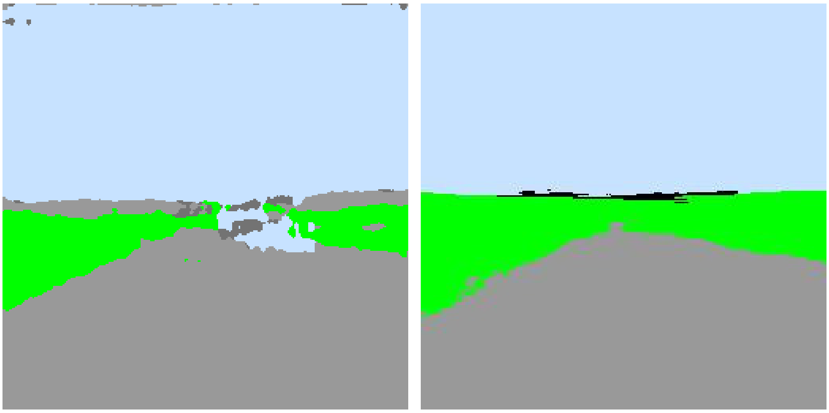
Figure 6.19: Region labeling approaches on Cityscapes dataset. (blue, light gray, darker gray, green and black indicate "sky", "road", "construction", "vegetation" and "unknown", respectively.)

6.7 Summary and Conclusions

In this section, we review four traffic surveillance use cases in which contextual information at several levels of a scene enables the automated analysis of the complicated scenarios that was previously not possible using conven-



(a) Image from LabelMe [107] (b) Manually annotated regions



(c) The deep learning-based region labeling of (a) (d) Our gravity-based region labeling of (a)

Figure 6.20: Region labeling approaches on the LabelMe dataset.

tional object classification techniques. Furthermore, we discuss the comparison between our gravity-based and a learning-based semantic region labeling approaches.

6.7.1 Summary

For smart surveillance systems, it is important to achieve a reliable scene understanding and, based on this, event detection and interpretation. This high level of understanding is enabled by incorporating context information from several levels of the scene, i.e., pixel, region/object, context and scene. The key objectives of this chapter are on applying the proposed contextual information extraction techniques from pixel and region levels, for a higher level of scene understanding or better object detection in video-based surveillance system. This chapter has demonstrated that context information is essential for decision making about the semantic understanding in surveillance video. For this purpose, several use cases were presented, such as moving ship detection in port surveillance, traffic action recognition, moving car detection from traffic surveillance videos and fast abnormal event detection.

In the ship detection use case, we have shown that combining motion contextual information with the semantic region labeling as a spatial contextual information improves the precision and recall of the proposed moving ship detection approach in comparison with the "Existing" [12] and "Cabin detector" [128] [110] approaches in harbor surveillance applications.

In the traffic action recognition use case, we have presented an approach which combines our semantic region labeling and automatic traffic sign information as semantic contextual information. We have shown that this combination improves the automatic decision making for detecting legal versus illegal traffic actions, such as crossing the road by pedestrians in dedicated zebra-crossing areas versus non-dedicated areas.

In the moving car detection use case, the semantic region labeling and salient region detection are both applied and combined with motion information. We have shown that this combination improves the accuracy and efficiency of the proposed approaches in comparison with another approach [132] as well as an approach where contextual information is not used [91].

In the last use case, a block-based approach is proposed to detect abnormal situations by analyzing the pixel-based motion context, as an alternative for the conventional object-based approach. We have shown that our proposed approach performs better compared to the approach proposed by Ihaddadene and Djeraba [106].

At the end of this chapter, we have compared our gravity-based region labeling approach with a recently developed deep learning-based system in order to provide insight into how our approach performs compared to emerging deep learning technology. The comparison results show that our gravity-based region labeling performs comparable or slightly lower to a deep learning-based region labeling system. By way of this comparison, we have demonstrated that our approach remains valuable even with the emerging of recently developed deep learning technologies, since it has a clearly lower complexity than a deep learning system.

6.7.2 Conclusion and discussion

In this chapter, we have shown that using contextual information enables the automated analysis of complicated scenarios that was previously not possible using conventional object classification techniques. Another conclusion is that context information is not complicated to extract from a surveillance video, while it adds significant value to the automated surveillance analysis.

The presented concepts of using context information at different levels for better automated scenario analysis are mostly novel, since they offer a higher level of understanding or a better robustness. This brings surveillance analysis generally at a higher level of quality, while improving the reliability of the video analysis at the scenario level. The frameworks are generic and do not depend on the type of scene, while the fast algorithms allow real-time execution. From the use-case studies, it can be concluded that context information is an important source for improving automated video surveillance analysis, as it not only improves the reliability of moving object detection (e.g. ships and cars), but also enables scene understanding that is far beyond object understanding (e.g. traffic scenarios). The experiments of this chapter prove this by demonstrating several surveillance scenarios where the semantic meaning of the events can only be detected due to the availability of the context. However, this conclusion is drawn with prudence. Unfortunately, the experiments based on scenarios with intelligent or advanced behavior of people/objects are based on a rather limited number of experiments, since such data is typically not available in shared datasets on the Internet. In some cases, such scenarios were generated by the research group with student actors. Therefore, although the experimental results are convincing, the related conclusions are only indicative and need a larger number of experiments, such that a real scoring percentage can be achieved after testing. It is estimated that with the growth of surveillance databases on the Internet, this large-scale testing of advanced behavioral analysis will soon be possible.

7

Conclusions

The concluding chapter of this thesis summarizes the contributions and conclusions of each individual chapter and provides answers to the research questions posed in Chapter 1. This chapter ends with an overall discussion on open issues and outlook on context-based surveillance video analysis.

7.1 Conclusions of the individual chapters

Chapter 2 has reviewed techniques for the image/video scene understanding. We found that the scene understanding research is mostly based on considering objects in isolation from the surrounding scene and contextual information is often not taken into account. Our review indicated that surveillance images/videos can be analyzed at different levels, i.e., pixel, region/object, context and scene level, for the purpose of scene understanding and event interpretation. We found that color, texture and motion are the most common features at the pixel level for image/video analysis. One of the most commonly used learning techniques is SVM which has been proven to be reliable and efficient. Moreover, our review indicated that features can be extracted in not only spatial and temporal domain but also in a given transformed domain. The common transform-based feature extraction techniques include DCT, FFT and Gabor filter.

Chapter 3 has introduced a number of simple, fast salient region detection techniques based on various features, such as sum of edge pixels, number of connected components, the number of straight lines found by the Hough transform and the entropy of DCT coefficients. Our experiments have shown that our DCT-based salient region detection approach provides better results compared to the other salient region detection techniques. We have compared our DCT-based salient region detection with Rahtu *et al.* [90]. The experimental results have shown that our DCT-based approach outperforms the recently published approach of Rahtu *et al.* with approximately 22%.

Chapter 4 has presented our research on region labeling for outdoor surveillance applications. Instead of making region-specific solutions, our research for semantic region labeling has resulted in developing a general framework for performing automatic semantic labeling task. Our generic region labeling framework is based on using spatial context for labeling regions. We have introduced two models for generic framework: (1) gravity-based and (2) Global Region Statistics (GRS)-based models. In our gravity-based system, following a segmentation stage, color, texture as well as the vertical position have been used to train a multiple-SVM classifier. In the GRS-based model the mean and standard deviation of the vertical region positions are exploited.

Experimental results have shown that our gravity-based model gives the best results and outperforms our GRS-based approach, the algorithms of Bao *et al.* and Millet *et al.* [109] with 2%, 2% and 3%, respectively. Our gravity-based approach is highly adaptive to new semantic region types because it avoids preset rules which can reduce flexibility. As a conclusion, our gravity-based generic region labeling approach does not need to re-design for each new region type as with specific region labeling.

Chapter 5 has analyzed the computational complexity calculations of our DCT-based salient region detection and gravity-based semantic region labeling approaches. It is proven that the algorithms execute with low complexity, to support real-time and embedded systems. The applied complexity analysis method has been based on counting intrinsic/native DSP operations, like Mega Operations per Frame (MOPF) or per second (MOPS). Regarding our DCT-based salient region detection approach, our complexity analysis has shown that it outperforms Rahtu's algorithm with 10 times lower complexity. We have shown that in Rahtu's algorithm the step of converting RGB-to-Lab color space has the highest contribution (88%) in the total complexity calculation (1404.75 MOPS for frame rate of 25 Hz) of the approach.

Our gravity-based labeling approach has been compared with a rule-based approach proposed by Millet et al. Our complexity analysis has shown that it outperforms Millet's algorithm with 24% lower complexity. This reduction in complexity is mainly because in our approach, the average number of support vectors has been 28% lower than that of Millet's approach and consequently, the complexity of the classification stage in our approach is decreased by 5.65 MOPF.

Chapter 6 has shown that using contextual information enables the automated analysis of complicated scenarios that was previously not possible using conventional object classification techniques.

In the ship detection scenario, the contextual information is provided by motion information, and our region labeling algorithm. Large advantages are that it detects the entire ship instead of only a part of the ship. Furthermore, the approach is able to handle occlusions between different ships and is robust to clutter caused by vegetation. The system is compared to two recent ship detection algorithms and shows robustness and good accuracy for real-life surveillance videos.

In the traffic action recognition scenario, the contextual information is provided by traffic sign detection classification system, and a region labeling algorithm. Using the contextual clues, the system is able to distinguish between people crossing a street at a zebra crossing and people crossing the street at a potentially dangerous location. It has been proven that the proposed traffic action recognition system evaluations works well for the dataset at hand due to the considerable testing of the individual components and the relative simplicity of the decision engines for the scenario content.

In the car detection scenario, in the first system, our gravity-based semantic region labeling algorithm is combined with motion information. We have shown that our approach performs better than the approach proposed by Wijnhoven *et al.* [132]. In contrast to our approach, the Wijnhoven's work erroneously detects several traffic signs as vehicles and misses occluded cars. In the second system, our DCT-based salient region detection technique is combined with a car detection approach. We have shown that although the detection rate is reduced, the proposed framework leads to applying the car detector, which is a computationally expensive algorithm, to only 9.7% of the total amount of image pixels. Therefore, the approach becomes more efficient.

In fast abnormal event detection scenario, we have developed a motion-context-based algorithm to detect abnormal events in surveillance videos of a public place. Our major contribution is introducing informative features based on motion and using an automatically updated threshold to detect abnormal events. We have discovered that the entropy of the DCT-transformed motion magnitude is a reliable measure for classifying whether the current activity in the video is normal or not. Our framework is generic and does not depend on the type of scene.

At the end of Chapter 6, we have compared our gravity-based region labeling with a recently developed region labeling system by Meletis *et al.* [2] which is based on emerging Deep Learning technology, tested on the Cityscapes and LabelMe datasets. We have found that on the Cityscapes dataset, both approaches show comparable results. The Deep Learning-based approach performs slightly higher for vegetation, construction and road classes. Using the LabelMe dataset, both approaches show comparable results. Our gravity-based approach presents slightly better results than the Deep Learning-based method for the road class, while the Deep Learning-based approach performs slightly higher than our gravity-based approach for the construction class. We conclude that the proposed algorithms perform on the average slightly lower than Deep Learning, albeit with a clearly lower computational complexity.

7.2 Discussion on research questions

We will now evaluate our proposed approaches with respect to each of the posed research questions in Section 1.3.

RQ1: *How do we define and categorize contextual information for outdoor surveillance video applications?*

Chapter 2 has described levels at which information can be exploited. These levels include pixel, region/object, context and scene levels. We have discussed that contextual information can be categorized at (1) pixel-level such as color, texture, edge, motion, DCT and Hough-based transform, (2) region-level, such as salient and semantic-labeled regions as background information, and (3) at the level of scene understanding such as semantically meaningful information for specific objects or behavior, etc. Furthermore, we have foound that in outdoor surveillance applications contextual information can be exploited from static background (road, sky, etc) and moving background (water). Additionally, information at multiple layers of regions are taken into account as side context. For example, zebra crossings and traffic signs are accounted for as layers of road regions.

RQ2a : *How salient regions detection and semantic region labeling can be used for surveillance applications?*

Chapter 3 and 4 have discussed that region analysis can be applied in two ways: (1) to detect an arbitrary region in an image for general purposes, or (2) to be used as a side or contextual information when it is inserted as a processing block in another larger algorithm or an application.

Chapter 3 has discussed that regions including human-made objects such as cars or suitcases are more informative. This is because objects that are often involved in dangerous situations are made by human and therefore are potentially more suspicious for outdoor video surveillance applications. Regions consisting human-made objects include of high amount of lines, edges or any other high frequency information. Techniques such as Sobel edge detection, Hough-based line detection and DCT-based feature extraction approaches can be used to identify such salient regions. Therefore, it is feasible to analyze scenes explicitly within salient regions for surveillance applications.

Chapter 4 has aimed at developing a general framework for performing automatic semantic region labeling task. We have introduced a generic gravitational framework based on spatial context (in our case vertical position information) for labeling regions. This model is based on the observation that each region is more likely to be found at a specific vertical position. We have found that this gravitational model is attractive because it allows distinguishing between regions with similar color and texture, which are the most common features, based on their vertical positions. For example, sky region which has similar color and texture as water region, often occurs at a higher vertical position compared to water regions.

RQ2b : *Which approaches perform better in terms of accuracy for salient region detection and for semantic region labeling?*

Chapter 3 has shown that the Hough- and DCT-based methods performs similarly and are better than both edge and connected component-based methods in terms of recall and precision. However, the Hough-based method is the slowest algorithm. Therefore, we have concluded that the DCT-based approach performs better compared to the other approaches.

Chapter 4 has shown that our gravity-based model gives the best results and outperforms other approaches. The gravity-based generic region labeling approach has shown more promising results compared to our semantic labeling of specific regions because it does not need to be redesigned for each new region type as for specific region labeling. It also outperforms our GRS-based semantic region labeling.

RQ3a : *How computational complexity of algorithms is estimated?*

Chapter 5 has discussed a metric for estimating the computational complexity based on Operations Per Frame (OPF), Mega Operations Per Frame (MOPF) or Per second (MOPS). These metrics are based on counting native DSP operations, like multiplications, additions, data storing and loading. By incorporating data storing and loads, also an indication of the memory usage and bandwidth is obtained. We do not rely on execution time because it also depends on the platform and programming environment on which the algorithms are implemented.

RQ3b : *Are salient region detection and semantic region labeling methods feasible for (near) real-time applications with respect to complexity and do they differ from available methods in the literature in terms of computational complexity?*

Chapter 5 has shown that our DCT-based salient region detection and gravity-based region labeling techniques have lower computational complexity compared to available methods in the literature. The computational complexity analysis also indicates lower memory usage and bandwidth. Therefore, our approaches are more feasible compared to the studied approaches for (near) real-time applications.

RQ4a : *Are detecting salient regions and labeling of regions with semantically meaningful labels feasible for practical surveillance applications?*

Chapter 6 has shown that using contextual information in traffic surveillance applications increases the reliability of the detection of objects (e.g. ships in harbor and cars in street surveillance videos), which helps to achieve high level of surveillance scene understanding. Furthermore, this chapter has discussed that using contextual information enables the automated analysis of complicated traffic surveillance scenarios (e.g. abnormal traffic actions) that was previously not possible using conventional object classification techniques. The work does open new possibilities for more intelligent surveillance applications.

RQ4b : *In which cases and scenarios using contextual information contributes to obtain more reliable and robust surveillance systems in practice?*

We have studied different applications in Chapter 6 in which the contextual information have contributed for obtaining a more reliable and robust surveillance scene understanding. Two examples of such case are: (1) *Detection of moving ship in harbor surveillance*: where, our semantic region labeling approach introduced in Chapter 4 is combined with motion context information to detect moving ships in port surveillance videos, and (2) *Recognition of traffic action*: where, our semantic region labeling algorithm proposed in Chapter 4 and automatic traffic sign information [3] are combined to recognize actions in traffic surveillance videos. In the first scenario, the detection of the water region as semantic region can help detecting ships, because ships can only travel within the water region with a salient motion. The concept of the salient motion is based on the common understanding that ships should have significant motion. In the traffic action recognition scenario, various traffic actions are more completely understood by the system when using contextual clues. Using traffic signs and zebra crossing regions as the contextual clues, the system is able to distinguish between people crossing a street normally and in a dangerous fashion.

7.3 Overall discussion and outlook

This thesis has presented context-based automatic video surveillance systems. Automatic video understanding has a demanding but essential applications in the video surveillance domain. We have argued that by adding contextual information about objects and/or scenes, a better classification and understanding is achieved. We have shown that contextual information is not complicated to extract from a surveillance video, while it adds significant value to the automated surveillance analysis. Using contextual information enables the automated analysis of complicated scenarios that was previously not possible using conventional object classification techniques. The presented concepts of using context information at different levels for better automated scenario analysis are mostly novel and offer a higher level of understanding or a better robustness. This brings surveillance analysis generally at a higher level of quality, while improving the reliability of the video analysis at the scenario level. The proposed algorithms are designed to be embedded as a processing block in another larger algorithm or another application (e.g. labelled water region as context to be used in an advanced ship detection algorithm). Moreover, the discussed algorithms and techniques are not limited to static cameras, and enable their use also on moving cameras. Although the individual components of the presented complex scenarios were broadly tested for multiple datasets, the number of experiments at the scenario level were rather limited. This is mainly due to the lack of test material relevant to our video surveillance applications. When more relevant test data become available, which is now rapidly developing due to the growth of sharing data for Deep Learning experiments, future studies can be based on more elaborate experiments on advanced behavioral analysis.

An element of discussion here is whether higher scores for reliable object detection should be based on the explicit use of context information. With the current growth of Deep Learning solutions and learning-based techniques, already a significant performance improvement has been realized, solely by employing these new techniques and the involved large-scale learning of object instances. Nevertheless, we consider the concept of context information to remain valuable for specific cases, especially for advanced semantic event analysis.

Here, we have taken steps to provide a generic frameworks that can handle variations in a given scene and do not depend on the type of scene. In our research for semantic region labeling, we have considered five semantic region types, but this number can be extended so that more semantic region types are explored. Furthermore, in our research for DCT-based salient region detection algorithm we have considered frequency features which can be affected by the compression algorithm and signal-to-noise ratio. Therefore, by adding additional features that are not affected by noise this limitation can be overcome.

In addition to satisfying accuracy, efficient automated video surveillance systems are required. As such, improvements are needed in terms of providing algorithms with lower computational complexity for (near) real-time image analysis, or very efficient computing architectures, or a combination of both. In this thesis, we have discussed an approach for calculating computational complexity that is based on counting native DSP operations and memory storage actions with a basic RISC CPU as a reference model. We have shown that this complexity estimation technique applies to the field of video surveillance and explicitly provides useful insights into computational complexity.

However, with the strong developments of convolutional neural networks (CNNs), such complexity analysis has to be performed on e.g. the scaling and filtering layers of the network and the fully connected layers. It is interesting to see that some of the recent novel architectures for CNNs like Darknet and Squeezenet are emerging for low-cost applications. Besides this, computing cores like GPUs and programmable device makers now also develop partly dedicated small-sized and low-power solutions, specifically intended for embedded systems. Therefore, it is only a matter of time before CNN-based analysis systems will enter the surveillance domain and may even grow into surveillance cameras in the near future. We consider that some of the concepts can translate into these new rapidly evolving Deep Learning-based systems.

Similar fast technology developments take place for visual sensing, where e.g. infrared (thermal) sensing can be used for better context extraction and content analysis and thereby improve automated video surveillance systems in better decision making. Comparable reasoning holds for multispectral sensing, which could also contribute to safer surveillance systems. In all such cases, Deep Learning technology can be successfully applied, however, the surveillance market itself will again impose its cost and complexity limitations on such advanced sensing solutions. Perhaps the largest need at present for surveillance and automotive video systems is that the efficiency of CNN-based and learning-based systems is more carefully considered than has been done up to this point, to facilitate and sustain the broad use of high-efficiency CNN-based video analysis systems for the next generation smart camera and surveillance systems.

Complete Bibliography

[1] Oge Marques, Elan Barenholtz, and Vincent Charvillat. “Context modeling in computer vision: techniques, implications, and applications”. In: *Multimedia Tools and Applications* 51.1 (2011), pp. 303–339.

[2] Panagiotis Meletis and Gijs Dubbelman. “Training of Convolutional Networks on Multiple Heterogeneous Datasets for Street Scene Semantic Segmentation”. In: *arXiv preprint arXiv:1803.05675* (2018).

[3] Lykele Hazelhoff, Ivo M. Creusen, and Peter H.N. de With. “Robust classification system with reliability prediction for semi-automatic traffic-sign inventory systems”. In: *IEEE Workshop on the Applications of Computer Vision (WACV)*. 2013.

[4] Solmaz Javanbakhti, Xinfeng Bao, Svitlana Zinger, et al. “Automatic generic Region-Of-Interest selection for video surveillance applications”. In: (2014), p. 97.

[5] Ivo M Creusen and Lykele B Hazelhoff. “Automatic recognition system for surveying of traffic signs and road markings from street-level panoramic images”. In: (2016).

[6] Solmaz Javanbakhti, Xinfeng Bao, Ivo Creusen, et al. “Adding Context Information to Video Analysis for Surveillance Applications”. In: *Emerging Research on Networked Multimedia Communication Systems* (2015), p. 159.

[7] Alexei A Efros, Alexander C Berg, Greg Mori, et al. “Recognizing Action at a Distance.” In: *ICCV*. Vol. 3. 2003, pp. 726–733.

[8] David Minnen, Irfan Essa, and Thad Starner. "Expectation grammars: Leveraging high-level expectations for activity recognition". In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2003, pp. II–II.

[9] Vasu Parameswaran and Rama Chellappa. "View invariance for human action recognition". In: *International Journal of Computer Vision* 66.1 (2006), pp. 83–101.

[10] Oswald Lanz. "Approximate bayesian multibody tracking". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.9 (2006), pp. 1436–1449.

[11] Loris Bazzani, Marco Cristani, and Vittorio Murino. "Decentralized particle filter for joint individual-group tracking". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 1886–1893.

[12] Xinfeng Bao, Svitlana Zinger, Rob GJ Wijnhoven, et al. "Ship detection in port surveillance based on context and motion saliency analysis". In: *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics. 2013, pp. 86630D–86630D.

[13] Carolina Galleguillos and Serge Belongie. "Context based object categorization: A critical survey". In: *Computer vision and image understanding* 114.6 (2010), pp. 712–722.

[14] Santosh K Divvala, Derek Hoiem, James H Hays, et al. "An empirical study of context in object detection". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 1271–1278.

[15] Aude Oliva and Antonio Torralba. "Modeling the shape of the scene: A holistic representation of the spatial envelope". In: *International journal of computer vision* 42.3 (2001), pp. 145–175.

[16] Derek Hoiem, Alexei A Efros, and Martial Hebert. "Recovering surface layout from an image". In: *International Journal of Computer Vision* 75.1 (2007), pp. 151–172.

[17] John R Smith and Shih-Fu Chang. "Single color extraction and image query". In: *Image processing, 1995. Proceedings., International conference on*. Vol. 3. IEEE. 1995, pp. 528–531.

[18] Dengsheng Zhang, Md Monirul Islam, and Guojun Lu. "A review on automatic image annotation techniques". In: *Pattern Recognition* 45.1 (2012), pp. 346–362.

[19] Anil K Jain and Aditya Vailaya. "Image retrieval using color and shape". In: *Pattern recognition* 29.8 (1996), pp. 1233–1244.

[20] Myron Flickner, Harpreet Sawhney, Wayne Niblack, et al. "Query by image and video content: The QBIC system". In: *computer* 28.9 (1995), pp. 23–32.

[21] Greg Pass and Ramin Zabih. "Histogram refinement for content-based image retrieval". In: *Applications of Computer Vision, 1996. WACV'96., Proceedings 3rd IEEE Workshop on*. IEEE. 1996, pp. 96–102.

[22] Jing Huang, S Ravi Kumar, Mandar Mitra, et al. "Image indexing using color correlograms". In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE. 1997, pp. 762–768.

[23] David Marr and Ellen Hildreth. "Theory of edge detection". In: *Proceedings of the Royal Society of London B: Biological Sciences* 207.1167 (1980), pp. 187–217.

[24] John Canny. "A computational approach to edge detection". In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), pp. 679–698.

[25] Manfred H Hueckel. "An operator which locates edges in digitized pictures". In: *Journal of the ACM (JACM)* 18.1 (1971), pp. 113–125.

[26] Raman Maini and Himanshu Aggarwal. "Study and comparison of various image edge detection techniques". In: *International journal of image processing (IJIP)* 3.1 (2009), pp. 1–11.

[27] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. "Textural features corresponding to visual perception". In: *IEEE Transactions on Systems, Man, and Cybernetics* 8.6 (1978), pp. 460–473.

[28] Rafael C Gonzalez. *woods, RE and Eddins, SL,"Digital Image Processing using MATLAB"*. 2002.

[29] Fuhui Long, Hongjiang Zhang, and David Dagan Feng. "Fundamentals of content-based image retrieval". In: *Multimedia Information Retrieval and Management*. Springer, 2003, pp. 1–26.

[30] Md Monirul Islam, Dengsheng Zhang, and Guojun Lu. "A geometric method to compute directionality features for texture images". In: *Multimedia and Expo, 2008 IEEE International Conference on*. IEEE. 2008, pp. 1521–1524.

[31] Xing-Jian He, Yue Zhang, Tat-Ming Lok, et al. "A new feature of uniformity of image texture directions coinciding with the human eyes perception". In: *International Conference on Fuzzy Systems and Knowledge Discovery*. Springer. 2005, pp. 727–730.

[32] Sos S Agaian, Blair Silver, and Karen A Panetta. "Transform coefficient histogram-based image enhancement algorithms using contrast entropy". In: *IEEE transactions on image processing* 16.3 (2007), pp. 741–758.

[33] Tienwei Tsai, Yo-ping Huang, and Te-Wei Chiang. "Image retrieval based on dominant texture features". In: *Industrial Electronics, 2006 IEEE International Symposium on*. Vol. 1. IEEE. 2006, pp. 441–446.

[34] K Ramamohan Rao and Ping Yip. *Discrete cosine transform: algorithms, advantages, applications*. Academic press, 2014.

[35] K Kanagalakshmi and E Chandra. "Frequency Domain Enhancement algorithm based on Log-Gabor Filter in FFT Domain". In: *European Journal of Scientific Research* 74.4 (2012), pp. 563–573.

[36] Joni Kamarainen, Ville Kyrki, and H Kalviainen. "Fundamental frequency Gabor filters for object recognition". In: *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. Vol. 1. IEEE. 2002, pp. 628–631.

[37] Itzhak Fogel and Dov Sagi. "Gabor filters as texture discriminator". In: *Biological cybernetics* 61.2 (1989), pp. 103–113.

[38] Meng Yang, Lei Zhang, Simon Chi-Keung Shiu, et al. "Monogenic binary coding: An efficient local feature extraction approach to face recognition". In: *IEEE Transactions on Information Forensics and Security* 7.6 (2012), pp. 1738–1751.

[39] David J Field. "Relations between the statistics of natural images and the response properties of cortical cells". In: *JOSA A* 4.12 (1987), pp. 2379–2394.

[40] Richard O Duda and Peter E Hart. "Use of the Hough transformation to detect lines and curves in pictures". In: *Communications of the ACM* 15.1 (1972), pp. 11–15.

[41] Laurence Likforman-Sulem, Anahid Hanimyan, and Claudie Faure. "A Hough based algorithm for extracting text lines in handwritten documents". In: *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*. Vol. 2. IEEE. 1995, pp. 774–777.

[42] Weiming Hu, Tieniu Tan, Liang Wang, et al. "A survey on visual surveillance of object motion and behaviors". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.3 (2004), pp. 334–352.

[43] Nasim Arshad, Kwang-Seok Moon, and Jong-Nam Kim. "Multiple ship detection and tracking using background registration and morphological operations". In: *Signal Processing and Multimedia*. Springer, 2010, pp. 121–126.

[44] Alberto Broggi, Andrea Cappalunga, Stefano Cattani, et al. "Lateral vehicles detection using monocular high resolution cameras on TerraMax$^{\text{TM}}$". In: *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE. 2008, pp. 1143–1148.

[45] Birgi Tamersoy and Jake K Aggarwal. "Robust vehicle detection for tracking in highway surveillance videos using unsupervised learning". In: *Advanced Video and Signal Based Surveillance, 2009. AVSS'09. Sixth IEEE International Conference on*. Ieee. 2009, pp. 529–534.

[46] Yuan-Kai Wang and Shao-Hua Chen. "A robust vehicle detection approach". In: *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference on*. IEEE. 2005, pp. 117–122.

[47] Sepehr Aslani and Homayoun Mahdavi-Nasab. "Optical flow based moving object detection and tracking for traffic surveillance". In: *International Journal of Electrical, Electronics, Communication, Energy Science and Engineering* 7.9 (2013), pp. 789–793.

[48] Hai-Yan Zhang. "Multiple moving objects detection and tracking based on optical flow in polar-log images". In: *Machine Learning and Cybernetics (ICMLC), 2010 International Conference on*. Vol. 3. IEEE. 2010, pp. 1577–1582.

[49] Ali Wali and Adel M Alimi. "Event detection from video surveillance data based on optical flow histogram and high-level feature extraction". In: *2009 20th International Workshop on Database and Expert Systems Application*. IEEE. 2009, pp. 221–225.

[50] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. "Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods". In: *International Journal of Computer Vision* 61.3 (2005), pp. 211–231.

[51] Simon Hartmann Have, Huamin Ren, and Thomas B Moeslund. "Real-time Multiple Abnormality Detection in Video Data". In: *VISAPP 2013* (2013).

[52] Ali Borji and Laurent Itti. "State-of-the-art in visual attention modeling". In: *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2013), pp. 185–207.

[53] Anne M Treisman and Garry Gelade. "A feature-integration theory of attention". In: *Cognitive psychology* 12.1 (1980), pp. 97–136.

[54] Christof Koch and Shimon Ullman. "Shifts in selective visual attention: towards the underlying neural circuitry". In: *Matters of intelligence*. Springer, 1987, pp. 115–141.

[55] Josep M Gonfaus, Xavier Boix, Fahad S Khan, et al. "Harmony potentials: Fusing global and local scale for semantic image segmentation". In: (2010).

[56] Stefan Kluckner, Thomas Mauthner, Peter M Roth, et al. "Semantic Image Classification using Consistent Regions and Individual Context." In: *BMVC*. Vol. 6. 2009, p. 7.

[57] Lubor Ladicky, Chris Russell, Pushmeet Kohli, et al. "Graph cut based inference with co-occurrence statistics". In: *European Conference on Computer Vision*. Springer. 2010, pp. 239–253.

[58] Vivek Dey, Yun Zhang, and Ming Zhong. "A review on image segmentation techniques with remote sensing perspective". In: (2010).

[59] Jianbo Shi and Jitendra Malik. "Normalized cuts and image segmentation". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[60] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural networks* 61 (2015), pp. 85–117.

[61] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[62] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[63] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[64] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[65] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, et al. “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2018), pp. 834–848.

[66] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[67] Andrew G Howard, Menglong Zhu, Bo Chen, et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv preprint arXiv:1704.04861* (2017).

[68] Joseph Redmon. “Darknet: Open source neural networks in c”. In: (2013).

[69] Forrest N Iandola, Song Han, Matthew W Moskewicz, et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size”. In: *arXiv preprint arXiv:1602.07360* (2016).

[70] Yali Amit and Donald Geman. *Randomized Inquiries About Shape: An Application to Handwritten Digit Recognition.* Tech. rep. DTIC Document, 1994.

[71] Yali Amit and Donald Geman. “Shape quantization and recognition with randomized trees”. In: *Neural computation* 9.7 (1997), pp. 1545–1588.

[72] Antonio Criminisi, Jamie Shotton, Ender Konukoglu, et al. “Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning”. In: *Foundations and Trends® in Computer Graphics and Vision* 7.2–3 (2012), pp. 81–227.

[73] Rob GJ Wijnhoven. “Object categorization and detection and their application in surveillance”. PhD thesis. Technische Universiteit Eindhoven, 2013.

[74] Olivier Chapelle, Patrick Haffner, and Vladimir N Vapnik. “Support vector machines for histogram-based image classification”. In: *IEEE transactions on Neural Networks* 10.5 (1999), pp. 1055–1064.

[75] Giorgos Mountrakis, Jungho Im, and Caesar Ogole. “Support vector machines in remote sensing: A review”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.3 (2011), pp. 247–259.

[76] Pakorn Watanachaturaporn, Manoj K Arora, and Pramod K Varshney. “Multisource Classification Using Support Vector Machines”. In: *Photogrammetric Engineering & Remote Sensing* 74.2 (2008), pp. 239–246.

[77] AHR Rob Albers. “Modeling and control of image processing for interventional X-ray”. PhD thesis. Technische Universiteit Eindhoven, 2010.

[78] Hae Jong Seo and Peyman Milanfar. “Nonparametric bottom-up saliency detection by self-resemblance”. In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2009, pp. 45–52.

[79] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, et al. “Global contrast based salient region detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.3 (2015), pp. 569–582.

[80] Laurent Itti, Christof Koch, Ernst Niebur, et al. “A model of saliency-based visual attention for rapid scene analysis”. In: *IEEE Transactions on pattern analysis and machine intelligence* 20.11 (1998), pp. 1254–1259.

[81] Martin Jagersand. “Saliency maps and attention selection in scale and spatial coordinates: An information theoretic approach”. In: *Computer Vision, 1995. Proceedings., Fifth International Conference on*. IEEE. 1995, pp. 195–202.

[82] Allam Shehata Hassanein, Sherien Mohammad, Mohamed Sameer, et al. “A survey on Hough transform, theory, techniques and applications”. In: *arXiv preprint arXiv:1502.02160* (2015).

[83] Yun Zhai and Mubarak Shah. “Visual attention detection in video sequences using spatiotemporal cues”. In: *Proceedings of the 14th ACM international conference on Multimedia*. ACM. 2006, pp. 815–824.

[84] Jian Li, Martin D Levine, Xiangjing An, et al. “Visual saliency based on scale-space analysis in the frequency domain”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.4 (2013), pp. 996–1010.

[85] Stas Goferman, Lihi Zelnik-Manor, and Ayellet Tal. “Context-aware saliency detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.10 (2012), pp. 1915–1926.

[86] Qu Ying-Dong, Cui Cheng-Song, Chen San-Ben, et al. “A fast subpixel edge detection method using Sobel–Zernike moments operator”. In: *Image and Vision Computing* 23.1 (2005), pp. 11–17.

[87] Shane Torbert. *Applied computer science*. Springer, 2016.

[88] DS Guru, BH Shekar, and P Nagabhushan. "A simple and robust line detection algorithm based on small eigenvalue analysis". In: *Pattern Recognition Letters* 25.1 (2004), pp. 1–13.

[89] Piotr Dollar, Christian Wojek, Bernt Schiele, et al. "Pedestrian detection: An evaluation of the state of the art". In: *IEEE transactions on pattern analysis and machine intelligence* 34.4 (2012), pp. 743–761.

[90] Esa Rahtu, Juho Kannala, Mikko Salo, et al. "Segmenting salient objects from images and videos". In: *European Conference on Computer Vision*. Springer. 2010, pp. 366–379.

[91] Xinfeng Bao, Solmaz Javanbakhti, Svitlana Zinger, et al. "Context-based object-of-interest detection for a generic traffic surveillance analysis system". In: *Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on*. IEEE. 2014, pp. 136–141.

[92] Bahman Zafarifar and Peter H.N. de With. "Adaptive modeling of sky for video processing and coding applications". In: *in 27 th Symposium on Information Theory in the Benelux*. Citeseer. 2006.

[93] Shufang Liu and Peter H.N. de With. *Camera-based water region detection in harbor monitoring*. Tech. rep. 2010.

[94] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Context analysis: sky, water and motion". In: (2011).

[95] Andrew C Gallagher, Jiebo Luo, and Wei Hao. "Improved blue sky detection using polynomial model fit". In: *Image Processing, 2004. ICIP'04. 2004 International Conference on*. Vol. 4. IEEE. 2004, pp. 2367–2370.

[96] Stephen Herman and Erwin Bellers. "Locally-adaptive processing of television images based on real-time image segmentation". In: *Consumer Electronics, 2002. ICCE. 2002 Digest of Technical Papers. International Conference on*. IEEE. 2002, pp. 66–67.

[97] Larry H Matthies, Paolo Bellutta, and Mike McHenry. "Detecting water hazards for autonomous off-road navigation". In: *AeroSense 2003*. International Society for Optics and Photonics. 2003, pp. 231–242.

[98] Tuo-zhong Yao, Zhi-yu Xiang, and Ji-lin Liu. "Robust water hazard detection for autonomous off-road navigation". In: *Journal of Zhejiang University-SCIENCE A* 10.6 (2009), pp. 786–793.

[99] Arturo Rankin and Larry Matthies. "Daytime water detection based on color variation". In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE. 2010, pp. 215–221.

[100] Shengyan Zhou, Jianwei Gong, Guangming Xiong, et al. "Road detection using support vector machine based on online learning and evaluation". In: *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE. 2010, pp. 256–261.

[101] Luo-Wei Tsai, Jun-Wei Hsieh, Chi-Hung Chuang, et al. "Lane detection using directional random walks". In: *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE. 2008, pp. 303–306.

[102] Yong Zhou, Rong Xu, Xiaofeng Hu, et al. "A robust lane detection and tracking method based on computer vision". In: *Measurement science and technology* 17.4 (2006), p. 736.

[103] Vasileios Mezaris, Ioannis Kompatsiaris, and Michael G Strintzis. "Compressed domain object detection for video understanding". In: *Proc. Workshop on Image Analysis For Multimedia Interactive Services (WIAMIS)*. 2004.

[104] Frank Schmitt and Lutz Priese. "Sky Detection in CSC-segmented Color Images." In: *VISAPP (2)*. 2009, pp. 101–106.

[105] Bahman Zafarifar and Peter HN de With. "Blue Sky Detection for Content-based Television Picture Quality Enhancement". In: *International Conference on Consumer Electronics*. Citeseer. 2007, pp. 1–2.

[106] Nacim Ihaddadene and Chabane Djeraba. "Real-time crowd motion analysis". In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE. 2008, pp. 1–4.

[107] Bryan C Russell, Antonio Torralba, Kevin P Murphy, et al. "LabelMe: a database and web-based tool for image annotation". In: *International journal of computer vision* 77.1-3 (2008), pp. 157–173.

[108] Xinfeng Bao, Solmaz Javanbakhti, Svitlana Zinger, et al. "Context modeling combined with motion analysis for moving ship detection in port surveillance". In: *Journal of Electronic Imaging* 22.4 (2013), pp. 041114–041114.

[109] Christophe Millet, Isabelle Bloch, P Hede, et al. "Using relative spatial relationships to improve individual region recognition". In: *Integration of Knowledge, Semantics and Digital Media Technology, 2005. EWIMT 2005. The 2nd European Workshop on the (Ref. No. 2005/11099)*. IET. 2005, pp. 119–126.

[110] Xinfeng Bao, Svitlana Zinger, Rob GJ Wijnhoven, et al. "Water region detection supporting ship identification in port surveillance". In: *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer. 2012, pp. 444–454.

[111] Pedro F Felzenszwalb and Daniel P Huttenlocher. "Efficient graph-based image segmentation". In: *International Journal of Computer Vision* 59.2 (2004), pp. 167–181.

[112] Csaba Benedek and Tamás Szirányi. "Study on color space selection for detecting cast shadows in video surveillance". In: *International Journal of Imaging Systems and Technology* 17.3 (2007), pp. 190–201.

[113] Guillaume Rellier, Xavier Descombes, Frederic Falzon, et al. "Texture feature analysis using a Gauss-Markov model in hyperspectral image classification". In: *IEEE Transactions on Geoscience and Remote Sensing* 42.7 (2004), pp. 1543–1551.

[114] Jamie Shotton, John Winn, Carsten Rother, et al. "Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context". In: *International Journal of Computer Vision* 81.1 (2009), pp. 2–23.

[115] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Fast sky and road detection for video context analysis". In: (2012).

[116] Arpit Jain, Abhinav Gupta, and Larry S Davis. "Learning what and how of contextual models for scene labeling". In: *European conference on computer vision*. Springer. 2010, pp. 199–212.

[117] Martin AR Pieck, Fons van der Sommen, Svitlana Zinger, et al. "Real-time semantic context labeling for image understanding". In: *Image Processing (ICIP), 2015 IEEE International Conference on*. IEEE. 2015, pp. 3180–3184.

[118] Abdiansah Abdiansah and Retantyo Wardoyo. "Time Complexity Analysis of Support Vector Machines (SVM) in LibSVM". In: *International Journal Computer and Application* (2015).

[119] Paulo Mateus, Daowen Qiu, and Lvzhou Li. "On the complexity of minimizing probabilistic and quantum automata". In: *Information and Computation* 218 (2012), pp. 36–53.

[120] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, et al. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012.

[121] Zhongde Wang. "Fast algorithms for the discrete W transform and for the discrete Fourier transform". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32.4 (1984), pp. 803–816.

[122] Mateus Beck Fonseca, João Baptista S Martins, and Eduardo A César da Costa. "Design of pipelined butterflies from Radix-2 FFT with Decimation in Time algorithm using efficient adder compressors". In: *Circuits and Systems (LASCAS), 2011 IEEE Second Latin American Symposium on*. IEEE. 2011, pp. 1–4.

[123] Amanpreet Kaur and BV Kranthi. "Comparison between YCbCr color space and CIELab color space for skin color segmentation". In: *IJAIS* 3.4 (2012), pp. 30–33.

[124] Xinfeng Bao, Solmaz Javanbakhti, Svitlana Zinger, et al. "Moving ship detection based on context modeling and motion analysis". In: *conference; Netherlands Conference on Computer Vision; 2014-04-24; 2014-04-25*. 2014.

[125] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Region labeling for surveillance video: techniques and application". In: *conference; Netherlands Conference on Computer Vision; 2014-04-24; 2014-04-25*. 2014.

[126] Ivo M Creusen, Solmaz Javanbakhti, Marijn JH Loomans, et al. "ViCoMo: visual context modeling for scene understanding in video surveillance". In: *Journal of Electronic Imaging* 22.4 (2013), pp. 041117–041117.

[127] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Fast abnormal event detection from video surveillance". In: *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp). 2012, p. 1.

[128] Rob GJ Wijnhoven, Kris van Rens, Egbert GT Jaspers, et al. "Online learning for ship detection in maritime surveillance". In: *Proc. of 31th Symposium on Information Theory in the Benelux*. Citeseer. 2010, pp. 73–80.

[129] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 886–893.

[130] Carolina Garate, Piotr Bilinsky, and François Bremond. "Crowd event recognition using hog tracker". In: *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*. IEEE. 2009, pp. 1–6.

[131] Antoni B Chan, Mulloy Morrow, and Nuno Vasconcelos. "Analysis of crowded scenes using holistic properties". In: *Performance Evaluation of Tracking and Surveillance workshop at CVPR*. 2009, pp. 101–108.

[132] Rob GJ Wijnhoven and PHN de With. "Fast training of object detection using stochastic gradient descent". In: *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE. 2010, pp. 424–427.

[133] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2001, pp. I–511.

[134] *cars 2001(Rear), Caltech*. `http://www.vision.caltech.edu/archive.html`. Accessed: 2014-04-30.

[135] Nick Kingsbury. "The Haar Transform". In: (2005).

[136] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. "Wider or deeper: Revisiting the resnet model for visual recognition". In: *arXiv preprint arXiv:1611.10080* (2016).

[137] Hui Yu, Mingjing Li, Hong-Jiang Zhang, et al. "Color texture moments for content-based image retrieval". In: *Image Processing. 2002. Proceedings. 2002 International Conference on*. Vol. 3. IEEE. 2002, pp. 929–932.

[138] Zi-Quan Hong. "Algebraic feature extraction of image for recognition". In: *Pattern recognition* 24.3 (1991), pp. 211–219.

[139] Syed Ali Khayam. "The discrete cosine transform (DCT): theory and application". In: *Michigan State University* 114 (2003).

[140] Cheng Li, Ivo Creusen, Lykele Hazelhoff, et al. "Detection and recognition of road markings in panoramic images". In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 448–458.

[141] Tony Lindeberg. "Discrete derivative approximations with scale-space properties: A basis for low-level feature extraction". In: *Journal of Mathematical Imaging and Vision* 3.4 (1993), pp. 349–376.

[142] Wei Wang, Jianwei Li, Feifei Huang, et al. "Design and implementation of Log-Gabor filter in fingerprint image enhancement". In: *Pattern Recognition Letters* 29.3 (2008), pp. 301–308.

[143] Sami Ekici, Selcuk Yildirim, and Mustafa Poyraz. "Energy and entropy-based feature extraction for locating fault on transmission lines by using neural network and wavelet packet decomposition". In: *Expert Systems with Applications* 34.4 (2008), pp. 2937–2944.

[144] Pooja Kamavisdar, Sonam Saluja, and Sonu Agrawal. "A survey on image classification approaches and techniques". In: *International Journal of Advanced Research in Computer and Communication Engineering* 2.1 (2013), pp. 1005–1009.

[145] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Context analysis: sky, water and motion". In: *International Workshop on Computer Vision Applications (CVA)* (2011), pp. 115–116.

[146] Solmaz Javanbakhti, Svitlana Zinger, and Peter HN de With. "Context-based region labeling for event detection in surveillance video". In: *Information Science, Electronics and Electrical Engineering (ISEEE), 2014 International Conference on*. Vol. 1. IEEE. 2014, pp. 94–98.

[147] Solmaz Javanbakhti, Svitlana Zinger, Rob GJ Wijnhoven, et al. "Fast scene analysis for surveillance & video databases". In: *IEEETransaction*. IEEE. 2017, pp. 136–141.

[148] Yu-Wen Huang, Ching-Yeh Chen, Chen-Han Tsai, et al. "Survey on block matching motion estimation algorithms and architectures with new results". In: *Journal of VLSI Signal Processing Systems* 42.3 (2006), pp. 297–320.

[149] Willem P Sanberg, Luat Do, et al. "Flexible multi-modal graph-based segmentation". In: *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer. 2013, pp. 492–503.

[150] Radhakrishna Achanta, Francisco Estrada, Patricia Wils, et al. "Salient region detection and segmentation". In: *International conference on computer vision systems*. Springer. 2008, pp. 66–75.

[151] Alexandre Winter, Henri Maitre, Nicole Cambou, et al. "Entropy and multiscale analysis: a new feature extraction algorithm for aerial images". In: *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*. Vol. 4. IEEE. 1997, pp. 2765–2768.

[152] Timor Kadir and Michael Brady. "Saliency, scale and image description". In: *International Journal of Computer Vision* 45.2 (2001), pp. 83–105.

[153] Sylvain Fischer, Filip Šroubek, Laurent Perrinet, et al. "Self-invertible 2D log-Gabor wavelets". In: *International Journal of Computer Vision* 75.2 (2007), pp. 231–246.

[154] Andrew Kirillov. "Motion detection algorithms". In: *The Code Project. Mar* (2007).

[155] Mohd Osman, Abdullah Zawawi Talib, Kian Lam Tan, et al. "Vehicle Monitoring System Using Motion Detection Algorithms For USM Campus." In: (2007).

[156] Nam Nguyen and Yunsong Guo. "Comparisons of sequence labeling algorithms and extensions". In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, pp. 681–688.

[157] John L Barron and Neil A Thacker. "Tutorial: Computing 2D and 3D optical flow". In: *Imaging Science and Biomedical Engineering Division, Medical School, University of Manchester* (2005).

[158] Wai Kin Kong, David Zhang, and Wenxin Li. "Palmprint feature extraction using 2-D Gabor filters". In: *Pattern recognition* 36.10 (2003), pp. 2339–2347.

[159] Claude L Fennema and William B Thompson. "Velocity determination in scenes containing several moving objects". In: *Computer graphics and image processing* 9.4 (1979), pp. 301–315.

[160] Simon Tong and Daphne Koller. "Support vector machine active learning with applications to text classification". In: *Journal of machine learning research* 2.Nov (2001), pp. 45–66.

Acknowledgements

I would like to express my sincere thanks and appreciation to all the people who helped and supported me during my PhD study. A special thank goes to my first promoter, Prof. Peter de With, for his guidance over the period that I was doing research at Video Coding and Architectures (VCA) group and thereafter when I was completing my thesis while working in the industry. He spent many late evenings to help improving my research output, including this thesis. I would also like to warmly thank my co-promoter Dr. Sveta Zinger for her supports during my PhD studies.

I am grateful to all the members of the promotion committee, for their time and effort in reading this thesis and their valuable feedback. I would like to thank prof.dr. Th. Gevers from University of Amsterdam, Prof. prof.dr.ir. R.N.J. Veldhuis from University of Twente, and Prof. prof.dr.ir. G. de Haan from Eindhoven University of Technology and dr.ir. R.G.J. Wijnhoven from ViNotion BV. I would like to particularly thank Dr. Wijnhoven for his detailed review and comments. I would also like to thank the chairman of my defense prof.dr.ir. A.M.J. Koonen.

I am also thankful of the support I received from my colleagues in VCA lab. Special thanks go to Anja and Marieke for all their supports and for bringing their positive energy into the group. Also, I acknowledge Panagiotis Meletis for collaborating in the analysis of Cityscapes dataSet, Martin A.R. Pieck for his contribution in real-time semantic region labeling and Bahman Zafarifar for his cotribution in this thesis.

It's my fortune to gratefully acknowledge the support of my friends with whom I had great time and enjoyable social life during my stay in the Netherlands/Europe; Gregory & Ifigenia, Sara, Davis, Francesco & Pein, Vale &

Manu, Marc and Lieke. I also appreciate all the kindness of my Persian friends together with them I never felt lonely and distanced from home; Ali & Niloofar, Hamid & Azar, Hossein & Samaneh, Ellaheh & Pooyan, Erfaneh & Ali, Raheleh & Adam, Elnaz & Arash, Shohreh, Leila & Salman, Nima & Judith, Hamed & Negar, etc.

I acknowledge the people who mean a lot to me, my parents, for showing faith in me and giving me liberty to choose what I desired. I salute you all for the selfless love, care, pain and sacrifice you did to shape my life. You were always willing to support any decision I made. I would never be able to pay back the love and affection showered upon by you. I am grateful to my sisters, Mozhgan and Sanaz for always being there for me. I am thankful for my beautiful nieces Paarmiss and Melody because they've shown me the innocence in the world.

I owe thanks to a very special person, my husband, Pooya for his continued and unfailing love, support and understanding during my pursuit of Ph.D degree that made the completion of thesis possible. You were always around at times I thought that it is impossible to continue, you helped me to keep things in perspective. I greatly value his contribution and deeply appreciate his belief in me. And on top of these are all the love I have for my little angle Lily-Rose. When I began writing this thesis you had not come to this world yet and I had not realized that babies grow quicker than books. I love you right up to the moon and back.

Curriculum vitae

Solmaz Javanbakhti was born in Tehran, Iran in 1981. She received her MSc degree in computer engineering in 2010 from Shahid Beheshti University, Tehran, Iran. Solmaz did her master thesis on shape modeling, skeletonization and segmentation of MR images at the Department of Biomedical Engineering at Eindhoven University of Technology. Following her master studies, Solmaz joined Video Coding and Architectures research group (VCA) at Eindhoven University of Technology, as a PhD researcher. Her research interests include video/image context analysis and region labeling for semantic understanding of scenes. Since 2014 Solmaz has been working in industry as a computer vision specialist in companies such as Bosch Security Systems in the Netherland as well as KLD Labs and ASML in the United States.

List of publications

The following conference and journal papers have been published based on the research presented in this thesis.

[1] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2017). Fast scene analysis for surveillance video databases. IEEE Transactions on Consumer Electronics, 63(3), 325-333. **This paper won a Chester Sall Best Paper Award which is annuel given for the best IEEE Transactions CE papers after a careful review- and selection procedure.**

[2] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2017). Fast semantic region analysis for surveillance & video databases. IEEE International Conference on Consumer Electronics (ICCE), 8-10 January, Las Vegas, USA.

[3] Javanbakhti, S., Bao, X., Creusen, I. M., Hazelhoff, L., Sanberg, W. P., Van de Wouw, D. W. J. M., Dubbelman, G., Zinger, S. & De With, P.H.N. (2015). Adding Context Information to Video Analysis for Surveillance Applications. Emerging Research on Networked Multimedia Communication Systems, 159.

[4] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2014). Context-based region labeling for event detection in surveillance video. In Information Science, Electronics and Electrical Engineering (ISEEE), International Conference on (Vol. 1, pp. 94-98). IEEE.

[5] Javanbakhti, S., Bao, X., Zinger, S. & De With, P.H.N. (2014). Automatic generic Region-Of-Interest selection for video surveillance appli-

cations. Proceedings of the 35th WIC Symposium on Information Theory in the Benelux and the 4th joint WIC/IEEE Symposium on Information Theory and Signal Processing in the Benelux, May 2-13 2014, Eindhoven, The Netherlands (pp. 97-104). Eindhoven: Technische Universiteit Eindhoven.

[6] Bao, X., Javanbakhti, S., Zinger, S., Wijnhoven, R.G.J. & De With, P.H.N. (2014). Context-based object-of-interest detection for a generic traffic surveillance analysis system. Proceedings of the 2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 26-29 August 2014, Seoul, South Korea (pp. 136-141).

[7] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2014). Region labeling for surveillance video: techniques and application. Netherlands Conference on Computer Vision (NCCV). Boekelo. Netherlands.

[8] Bao, X., Javanbakhti, S., Zinger, S. & De With, P.H.N. (2014). Moving ship detection based on context modeling and motion analysis. Netherlands Conference on Computer Vision (NCCV). Boekelo. Netherlands.

[9] Bao, X., Javanbakhti, S., Zinger, S., Wijnhoven, R. & De With, P.H.N. (2013). Context modeling combined with motion analysis for moving ship detection in port surveillance. Journal of Electronic Imaging, 22(4), 041114-041114.

[10] Creusen, I. M., Javanbakhti, S., Loomans, M. J., Hazelhoff, L. B., Roubtsova, N., Zinger, S. & De With, P.H.N. (2013). ViCoMo: visual context modeling for scene understanding in video surveillance. Journal of Electronic Imaging, 22(4), 041117-041117.

[11] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2012). Fast abnormal event detection from video surveillance. In Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV) (p. 1). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

[12] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2012). Fast sky and road detection for video context analysis. Oral : Proceedings of the 33rd WIC Symposium on Information Theory in the Benelux joint with the 2nd WIC/IEEE SP Symposium on Information Theory and Signal Processing in the Benelux, 24-25 May 2012, Boekeloo, The Netherlands, (pp. 210-218).

[13] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2011). Context analysis : sky, water and motion. Proceedings of the 32nd WIC Symposium on Information Theory in the Benelux, May 10-11, 2011, Brussels, Belgium.

[14] Javanbakhti, S., Zinger, S. & De With, P.H.N. (2011). Context analysis: sky, water and motion. International Workshop on Computer Vision Applications (CVA), (pp. 115-116). Netherlands.

[15] Javanbakhti, S., Moghadam, M.E., Hosseini, S.M. & Donkelaar, C.C. van (2010). Quantification of cartilage geometry using a skeletonization approach. Conference Paper : Proceedings of the 6th World Congress of Biomechanics (WCB 2010) 1-6 August 2010, Singapore, Singapore, (pp. SPKA00253-00433).