Doctoral Thesis

Scene understanding for mobile robots

Author(s):
Spinello, Luciano

Publication Date:
2009

Permanent Link:
https://doi.org/10.3929/ethz-a-006088392

Rights / License:
In Copyright - Non-Commercial Use Permitted
DISS. ETH. NO 18767

SCENE UNDERSTANDING FOR MOBILE ROBOTS

A dissertation submitted to
ETH ZURICH
for the degree of
Doctor of Sciences

presented by
LUCIANO SPINELLO
MSc Automation Engineering, Università di Roma Tor Vergata, Italy

11 February 1980
citizen of Italy

accepted on the recommendation of

Prof. Roland Y. Siegwart (ETH Zurich, thesis director)
Prof. James L. Crowley (INRIA Rhone-Alpes, examiner)
Dr. Kai O. Arras (Albert-Ludwigs-Universität Freiburg, examiner)

2009
Contents

Acknowledgments 9

Abstract 11

Abstract (Italian) 13

1 Introduction 15
  1.1 Motivations 16
  1.2 Original Contributions 18
  1.3 Outline of this thesis 20
    1.3.1 Chapter 2 20
    1.3.2 Chapter 3 20
    1.3.3 Chapter 4 20
    1.3.4 Chapter 5 20
    1.3.5 Appendix A 20
  1.4 Mathematical notations 21

2 Multimodal ROI Generation 23
  2.1 Related works 23
  2.2 Segmentation of Artefacts in Natural Environments 26
    2.2.1 Contributions 26
    2.2.2 Structure information processing 26
      2.2.2.1 Local plane fitting 27
      2.2.2.2 Smooth surfaces extraction 27
    2.2.3 Appearance information processing 28
      2.2.3.1 Color Clustering 29
      2.2.3.2 Color Quantization 30
    2.2.4 Information Fusion 30
      2.2.4.1 2.5D Self Organizing Network for clustering 32
    2.2.5 Experimental results 34
  2.3 Motion Segmentation Using a Local Entropy Field 39
    2.3.1 Contributions 39
3 Multimodal Object Detection

3.1 A Challenging Topic

3.2 State of The Art

3.3 Structure of the Chapter

3.4 Overview of the developed techniques

3.5 Contributions

3.6 Structure Information from Laser Data Analysis

3.6.1 Preparing Range Data: Clustering

3.6.1.1 Creating Segments

3.6.1.2 Reducing overclustering

3.6.2 Characterize Range Data: Feature Computation

3.6.3 Classifying Range Data: Feature Classification

3.6.3.1 Classifying range data with AdaBoost

3.6.3.2 AdaBoost with weighted soft SVMs

3.6.3.3 Classifying range data with Conditional Random Fields

3.6.4 Training and application of Range Classifiers

3.7 Range-based data detection evaluation

3.7.1 Quantitative measures of performance

3.7.2 Real World Datasets: Laser data

3.7.3 Evaluating of SBA classification for Pedestrian Detection

3.7.4 Evaluating of CRF classification for Multiclass Detection

3.8 Appearance Information from Image Data Analysis

3.8.1 Terminology: interest points and descriptors

3.8.2 Classifying Pedestrians with Histogram of Oriented Gradients

3.8.3 Classifying Pedestrians with Extended Implicit Shape Models

3.8.3.1 Generating a codebook

3.8.3.2 ISM Extension: Generating a Superfeatures Codebook

3.8.3.3 Generating object hypotheses

3.8.3.4 ISM Extension: Generating object hypotheses with Superfeatures Codebook

3.8.3.5 ISM Extension: Weighting the codebook entries and the shape flexibility

3.8.3.6 ISM Extension: High-dimensional Nearest Neighbor Search
CONTENTS

4.2.2 Related Work ............................................. 142
4.2.3 Overview .................................................. 143
4.2.4 Extraction of Mutually-Similar Object Instances ......... 144
  4.2.4.1 Object Extraction ..................................... 144
  4.2.4.2 Refining elements hypotheses ......................... 146
4.2.5 Analysis of Repetitive Objects ............................ 146
  4.2.5.1 Latticelets ............................................. 147
  4.2.5.2 Cycles and chains ..................................... 148
4.2.6 Structure Inference using Conditional Random Fields .... 148
  4.2.6.1 Node and Edge Features ............................... 149
  4.2.6.2 Network Structure ..................................... 149
4.2.7 Image Compression ........................................ 150
4.2.8 Experiments ............................................... 152
  4.2.8.1 Quantitative results ................................... 152
  4.2.8.2 Qualitative results ..................................... 152
4.3 Conclusions ................................................. 153

5 Conclusions ...................................................... 157
  5.1 Discussion .................................................. 157
    5.1.1 Chapter 2: Regions of Interest Generation Using Camera and Laser 157
    5.1.2 Chapter 3: Multimodal Object Detection in Urban Environments 158
    5.1.3 Chapter 4: Reasoning with Single Images .................... 159
  5.2 Future Works ................................................. 160
    5.2.1 Far to close range realtime robust multiclass detection . 160
    5.2.2 Large Scale Repetition Pattern Analysis .................. 160

A Smartter: A Mobile Urban Platform ................................ 161
  A.1 Vehicle description ........................................ 161
  A.2 Sensors ..................................................... 162
  A.3 Multimodal Sensors Configurations used ..................... 162
  A.4 Software architecture ...................................... 162

Curriculum Vitae .................................................. 165
  A.5 Biography ................................................... 165
  A.6 Awards ....................................................... 165
  A.7 Academic Projects .......................................... 166
  A.8 Academic Activities ........................................ 166
    A.8.1 List of publications and workshops ...................... 166
To my son Lorenzo,
my wife Francesca
and my dad.
Acknowledgements

At this important point of my life I would like to say thank you to all the people who accompanied and supported me during this work. In particular, I would like to thank professor Roland Siegwart, head of the Autonomous Systems Lab, without whom I would never have become a member of the lab. The interview I did with him in Lausanne in November 2005 was a turning point: I suddenly had the privilege of being accepted in an exciting scientific, international and multilingual environment, which I always dreamt since I was small child and I was yelling “I will be a robot scientist!! (or a fireman)” by breaking countless toys.

Furthermore, I would like to thank all the great people I met in the lab. Specially to Rudolph Triebel, he is a great scientist, methodical, rarely intelligent and, not secondary, a very good friend. I owe a lot to him, specially to the insights of classification and probabilistic techniques. We often shared the joy and the pain of working until dawn for completing a paper or being late at home for discussing novel ideas.

A special mention goes also to my swiss friend Stefan Gaechter, specially supportive during the beginning of the PhD and a sharp minded guy which you can talk about anything. I would also thank Krijstian Macek, with whom I shared the apartment for two years and we had great discussions, not only respect to robotics. We shared, along with the good Sasha Kolski, great Smart field experiences in Switzerland and in Germany.

Moreover I want to specially mention: Ralf (or as I call him: ‘Ralp’) Kästner and David Remy, simply great and funny guys. The good Dizan Vasquez for the honest discussions about robotics and various weird funny things. I would also to thank: Christian Bermes, for our great dueling in table soccer and for the fun we had together, Jiwon Shin, Jerome Maye, Martin Rufli, Viet Nguyen, Davide Scaramuzza. I would also thank Francesco and his brother Agostino Martinelli, without their recommendation letter I would never been interviewed by Roland.

A special mention goes to my thesis reviewers, prof. James L. Crowley and Dr. Kai Arras that contributed with very interesting comments to the improvement of this manuscript and to a very intense and interesting PhD defense.

I would like to thank the friendly guys of the Computer Vision Lab of ETH, specially prof. Bastian Leibe for the discussions regarding ISM; Andreas Ess, for the discussions regarding computer vision, the funny experiences and the hectoliters of beer we drank together in Japan.
I would like to thank my father, my sister and my brother for the support I always received from their positive attitude. If I am here is also because of them. I am also sure that my mother would be happy to know what I have done. I would also mention my closest and greatest friend Maurizio Fabriani and our countless chats about anything, from life facts to success in business. I want also thank my friend Francesco Bibbolino for the interesting discussions and for his taste in music.

Now, last but not least, a special *huge* thanks goes to Francesca, my girl and my love since forever, that always supported and did a lot for me. Lastly, an enormous kiss to Lorenzo, my beautiful little son, that will have an amazing future full of happiness and joy!
Abstract

All mobile robots operate in real environments, either indoors, for example in office environments, or outdoors, for example in urban like scenario. With time, more and more robots will participate and share the environment with humans, like for example autonomous cars or cleaning robots. Sharing the same space means also that a certain level of safety and interaction must be achieved. In order to create a robot that is intelligently aware of the space around itself, we need to increase its level of perception by performing an analysis of the surrounding scene. With this thesis we contribute to the field of scene analysis for mobile robots. In particular, this work addresses three big challenges: generating regions of interest, detecting and tracking objects in urban environments and reasoning on single images. This thesis is founded on multimodal approaches. Our work is focused on range and vision data processing. Each of these modality produce different but complementary kinds of information of the environment.

We produced probabilistic methods to generate regions of interest in range and camera data. We reduce the amount of information that has to be processed by the robot to focus only on salient regions. We presented two methods. The first method consist in a way of segmenting 3D range data and camera data to generate area of interest where artefacts could be found in natural environments. The second approach consist in the generation of regions of interest in an image sequence by clustering similar motions in the optical flow. These novel methods overcome the limitations of the current approaches, that tend to be computationally very expensive or need multiple data frames to work.

With this thesis, we present contributions in the field of multimodal detection and tracking in urban environments, namely pedestrian and car detection. We introduced a range based and image based detector that use machine learning techniques. They robustly encode geometrical and visual appearance of the objects. They are designed to be robust to occlusion and clutter typical of an urban scenario. Moreover, we introduce a tracking method that is used to fuse probabilistically the two information. With our approach we overcome the limitations of current camera-only and laser-only approaches: the requirement of enough contrast and good light conditions for camera-only methods and the low information content given by the scans of laser-only approaches.

Techniques to produce reasoning on single images represent the last contribution given by this thesis. We propose a generic method of discovering repetitive patterns found in an image. Thus, we are able to produce a compressed interpretation of an image and to infer missing parts. We used the paradigm of reasoning on single images for
producing a small scale pedestrian detector: it is based on classification of an effective low dimensional descriptor and a voting scheme. This method overcomes the usual requirement of robust interest point detectors and the usage of a scrolling detection window for detecting pedestrians.

The techniques presented in this thesis have been qualitatively and quantitatively evaluated on standard or real world datasets.
Abstract (Italian)

Tutti i robot mobili operano in ambienti reali, sia in interni, per esempio negli uffici, che in esterni, come nelle città. Con il tempo, sempre più robot parteciperanno e condividerranno l’ambiente con l’uomo, per esempio come nel caso delle automobili autonome o dei robot con compiti di pulizia. Condividere lo stesso ambiente vuol dire anche che è necessario un sufficiente livello di sicurezza e interazione. Per creare un robot che è intelligentemente consciolo dello spazio intorno a se, è necessario aumentare il suo livello di percezione compiendo un analisi della scena circostante. Con questa tesi diamo un contributo proprio al campo della comprensione della scena per robot mobili. In particolare, questa tesi affronta tre grandi sfide: la generazione di regioni di interesse, la detezione e l’inseguimento di oggetti in ambienti urbani, il ragionamento su singole immagini. Questa tesi si basa su approcci multimodali. L’intero lavoro è infatti principalmente basato sul processamento di dati di distanza, tramite sensori laser, e di immagini, tramite telecamere. Ognuna di queste modalità produce diversi ma complementari tipi di informazione.

Sono stati prodotti metodi probabilistici per generare regioni di interesse da dati di distanza e immagini. Si è ridotta la quantità di informazioni che deve essere processata dal robot, in modo da concentrare computazioni successive solo su parti salienti. I metodi presentati sono due. Il primo consiste in una tecnica per segmentare dati tridimensionali e immagini da telecamera, per generare aree di interesse dove è possibile localizzare manufatti umani in ambienti naturali. La seconda tecnica consiste nella generazione di regioni di interesse in una sequenza di immagini attraverso la segmentazione di moti simili nel flusso ottico. Questi metodi innovativi superano le limitazioni degli approcci correnti, che tendono ad essere computazionalmente molto esigenti o che necessitano di più campionamenti per ottenere una soluzione.

In questa tesi presentiamo contributi nel campo della detezione e inseguimento multimodale in ambienti urbani, specialmente sulla detezione di pedoni e automobili. Si è introdotto un detettore basato su dati di distanza e uno basato su immagini che utilizzano tecniche di machine learning. Tali detettori codificano la struttura geometrica e visuale degli oggetti. Sono stati progettati per essere robusti alle occlusioni e alle complessità tipiche di un ambiente urbano. Inoltre, è stato introdotto un metodo di inseguimento che è utilizzato per fondere probabilisticamente questi due tipi di informazioni. Il nostro approccio supera le limitazioni delle metodiche basate su dati solo visivi o solo di distanza. I dati immagine hanno dei requisiti minimi di contrasto e buona illuminazione, i dati laser hanno basso contenuto informativo.
L’ultimo contributo dato da questa tesi è rappresentato da tecniche per estrarre contenuto informativo da singole immagini. È proposto un metodo generale per rivelare modelli geometrici ripetitivi in immagini. Quindi è possibile produrre una interpretazione compressa di tale immagini e anche di predire parti mancanti. Oltretutto, è stato progettato un metodo per la detezione di pedoni di dimensioni molto piccole in immagini. Tale tecnica supera la necessità di punti di interesse robusti o dell’utilizzo di una finestra di detezione che deve essere computata per tutta la grandezza dell’immagine per ottenere detezione di pedoni.

Le tecniche presentate in questa tesi sono state valutate qualitativamente e quantitativamente su dati standard o su dati acquisiti in ambienti reali.
Introduction

Psychology and philosophy define the process of perception as the mechanism that elaborates sensorial data for attaining awareness. It is a very complex task of the human mind that intrigued scholars of these disciplines since centuries. Plato in the Allegory of the Cave (Plato [1992]) discusses about the difference between perceived reality and the truth. The Surangama Sutra (Hua [2009]) book in Buddhism contains meditations about reality, perception and the human mind. Descartes conceived perception as ordered sequence of events (Descartes [1990]): being surrounded, receiving sensory input, processing and producing a reaction. They say, perception is not just a mere form of reaction to sensory inputs but it is a rather intricate organization of sensory data in a complex experience, that is the final product of past knowledge, modeling capacity and interpretation sensibility of each being. In robotics, the richness level of human perception still represents a hard to reach dream: an illusion of self awareness for the machines. At this regard, this thesis puts effort in the interpretation of the sensory data, that layer of the perception reasoning that can be learned statistically through the data or through a modelization of the environment. The work contained in this thesis addresses the problem of scene analysis: the task of giving a meaning to the elements of the environment around the robot. The robot could then communicate and interact at a higher level form with humans, not anymore reporting raw data measurements but rich data labels, like "a pedestrian has been detected at this position", "a car is going in that direction".

Humans have amazing sensing capabilities, Aristotle, in his work De Anima (Aristotle [1987]), was the first to define a classification of human senses: sight, hearing, touch, smell, taste. Humans perform so well in understanding and analyzing the scene because they have multiple and complementary ways of retrieving data. As a matter of fact, children, during their first years, spend a long time exploring places and discovering objects by touching, looking and biting. Inspired by these observations, most of the works presented in this thesis follow a multiple sensor data processing approach. Our
idea is to combine, in the field of robotics, two of the most powerful human sensing capabilities, sight and touch. For robots this corresponds to use vision and range sensing in order to simultaneously obtain the appearance of the environment and its geometrical property.

Another impressive human talent is the capacity of interpretation in case of limited data quantity or the potential capabilities of memorizing only the salient information when perceiving a new environment. Similarly, with this thesis, we propose ways of reasoning on single images and ways of retrieving only repetitive information in images.

Precisely this thesis concentrates in proposing contributions and novelties in the creation of region of interest in sensory data, tracking and detection of objects in urban environments and in exploiting repetitive patterns of objects to improve the overall detection and compression.

1.1 Motivations

The general motivation behind all the works contained in this thesis is to increase the scene awareness of a mobile robot in urban environments.

A key ability for intelligent cars is autonomous navigation: the capacity of optimally planning a trajectory in an environment by taking into account obstacles and scenario constraints. Thus, such intelligent robots would greatly profit from an increased scene awareness to generate navigation plans that are sensitive to object categories, object trajectories and specific kinds of situations. Recent developments in autonomous urban navigation, like the DARPA urban challenge in 2007 (dar [2007]), fostered the development of reliable methods to negotiate complex traffic situations. Even though several teams successfully achieved the goal, the scenario proposed by the organizers was far from realistic. Only cars, driven by professional drivers, were allowed to be in the course. No pedestrians or other participants (motorcycles, bicycles) could cross the streets or move around. City guide robots need a high level of interaction: they must be aware of persons passing by, they need to track their positions, they need to plan a safe trajectory among the traffic of a city to reach a certain landmark. These two kinds of robotic applications have to face complicated situations that require much more reasoning than simple ‘reaction’ rules. The more information these robots are able to extract, the better the quality of the navigation, for the first case, and the quality of interaction with humans, in the latter.

Several challenges have been addressed by this thesis. A mobile robot, equipped with range sensors and cameras, is often overwhelmed by the amount of sensory information in outdoor scenario. Tens of information-rich image frames and range data segments are received each second: processing quickly such amount of data for detecting a moving object or for avoiding an obstacle could be problematic, specially in a crowded scenario. Thus, we defined a way of reducing the amount of information contained in the data by introducing methods to specify regions of interest. Our goal is to segment images or range data in regions that potentially contain only salient information. Therefore, common scene analysis tasks, like object detection and tracking, are simplified by processing only the interesting part of the data. Motivated by these reasons, we tackled the general problem
by using different techniques for two different sensor setups, commonly used in robotics: a camera-3D laser rangefinder setup and a single camera setup. For the first, we introduced a probabilistic method to segment images and 3D point clouds to highlight areas where artefacts could be found in natural environments. With the second setup, we define the process of regions of interest generation as a way of grouping coherent moving regions in an image sequence. Several methods have addressed this problem by defining motion-layers (Ayer and Sawhney [1995]), tensors (Tong et al. [2004]) or multibody analysis (Ke and Kanade [2001]). The drawback of these techniques consists in the requirement of several image frames to produce a solution or in the requirement of heavy computational resources. Instead, we tackled this problem by providing a two-frames only approach based on a novel measure of local optical flow similarity and we formulated the problem as a probabilistic clustering task.

Another difficult challenge that has been addressed in this thesis consist in the detection of very important objects in urban environments: pedestrians and cars. Literature shows that several robust methods exist to achieve this goal by using different sensors, among them camera based methods (Dalal and Triggs [2005], Leibe et al. [2005]) and laser based methods (Arras et al. [2007]) are the most diffuse and successful approaches. A real robotic system that shares the environment with humans have to efficiently function in different crowded situations, in various weather conditions and in various lightning conditions. Camera based methods, even though very robust, have the drawback of requiring a sufficient image contrast to work efficiently and of being dependent on the camera-lens for the size of the field of view. Large field of view lenses can cause heavy distortions and far away objects may appear very small. Instead, small field of view lenses are useful to magnify far objects but they produce just a narrow view of the surrounding environment. Laser rangefinder based methods have the drawback of obtaining just few points per scan, therefore objects are described by small sets of points. Lasers function without environment light and in virtually every weather condition. Motivated by these reasons, we overcome the problematics associated to each sensing modality by using a multimodal approach that combines an image based detector with a range based detector. We introduced several novelties in each kind detector: our image based detector uses sub-part reasoning, feature weighting, a smart cost function and multiclass capabilities. The range data detector takes in account local geometrical properties of the objects and also their neighborhood information, in order to obtain a robust detection in case of multiple objects categories. Moreover, we introduced tracking capabilities for obtaining a more reliable detection in time.

Vision is a sensor modality that has enormous potentials. Sometimes an image is everything a robot could retrieve to infer the solution of a certain problem. Far away pedestrians are hard to detect from single images: just few pixels define the shape and textures are washed out by the limited image resolution. Most of the methods present in literature use bags of features approaches (Leibe et al. [2005]) or detection window approaches (Viola et al. [2003]). The first methods are doomed by the lack of interests points, the second kind of techniques need to scroll a detection window through all the image area. Motivated by these reasons, we propose an hybrid method that overcomes
these two concepts by learning a novel small-scale image features and a pedestrian voting model. With this procedure we robustly learn which are the local small scale pedestrian characteristics that freely cast hypotheses for pedestrians in the image. Repetitive patterns analysis in images has been well studied in literature (Hays et al. [2006], Korah and Rasmussen [2008], Turina et al. [2001]). A little attention has been given to higher level interpretation of patterns, like repetition of windows in a building facade or the visual sequence of columns on a temple. Motivated by these reasons, we fill this void, and we propose a general solution to this problem by using an unsupervised method to segment and analyze such patterns. We then show how to use this information to infer missing elements and to provide an high level image compression.

These are the reasons why this thesis introduces novel techniques for detecting and tracking pedestrians and cars, for creating salient regions of interest in the environment, for reasoning on single images.

1.2 Original Contributions

A key effort of this work has been to bring together and to jointly propose novelties for two fields of robotics that are often considered separately: range based and camera based reasoning.

One contribution of this thesis is the segmentation of the environment by using different cues (motion, appearance) and different sensors (3D laser rangefinder, camera) to produce areas of interest, by using probabilistic reasoning. Structural information of 3D laser data and appearance information of camera data have been used to segment outdoor environments in order to highlight areas in which natural geometry and color are highly distinctive. Local smoothness and color uniformity are the properties taken in account as object cues that are processed for probabilistic 3D segmentation. A novel 2.5D Self Organizing Network definition has been used to segment likelihood from these two sensors. Another method, based only on vision, has been used for generating regions of interest based on a novel way to define consistency of the optical flow. We defined a measure of motion similarity called Local Entropy Field. It is a compact way of specifying local optical flow motion differences by taking into account, in small neighborhoods, magnitude, orientation and vector origin distances distributions. A Markov Random Field is run by taking this as an input for grouping vectors of similar motion. These works led to two publications: Spinello and Siegwart [2007, 2008b].

Another important contribution of this thesis lies in the detection and tracking of important urban objects, specially pedestrians and cars. Here a multimodal learning approach, based on laser and camera, has been used. For the image detector we introduced ISMe, an extension to the well known ISM object detection method (Leibe et al. [2005]). Our extended technique has multiclass capabilities; moreover, it uses objects subparts to define efficient hypotheses selection and it has advanced feature selection techniques. Laser range detector is based on Conditional Random Fields on the output of AdaBoost preclassification on geometrical and statistical features. Moreover we put efforts in combining the two informations together by using tracking with multiple motions. This work
led to four publications: Spinello and Siegwart [2008a], Spinello et al. [2008a,b, 2009b].

This thesis also proposes another contribution in the field of scene analysis by reasoning on single images. We present a novel method to exploit the arrangements of objects in an image based on geometrical probabilistic inference. We use this information to infer predictions, to include occurrences of weak detections and to achieve data compression. The image is subdivided in repetitive tokens that are analyzed for their spatial frequency. This information is represented in a compact form called \textit{latticelets}. They are minimal sets of repetitive elements and they are used to build the graph of Conditional Random Fields, to infer missing repetitive elements. Moreover, we also present an innovative technique to detect pedestrians at very small scale, by exploiting their segmented contour information with novel image descriptors. A pedestrian is detected by using a voting approach: each single classified segment casts votes for the center of the object. A large scale dataset has been used for reliably learning this method. This work led to a publication: Spinello et al. [2009a].

The work presented in this thesis is based on several publications in major conferences, journals and workshops. In the following, we group these by the corresponding chapters in the thesis.

- **Reasoning on multimodal data or optical flow to segment regions of interest (see specially Chapter 2)**
  
  
  

- **Multimodal detection and tracking in urban environments (see Chapter 3)**
  
  
  
  
  
  
  
1.3 Outline of this thesis

This thesis is organized in three main chapters. The first chapter focuses on generating interesting regions from images and/or range data. The second chapter explains the contribution given in the field of multimodal object detection and tracking. The third chapter presents how to exploit geometrical relations between objects in order to infer occurrences of weak detections and data compression. The outline of the thesis is the following.

1.3.1 Chapter 2

We here introduce a new approach for region of interest generation for mobile robotics. We describe a multimodal approach for segmenting outdoor environments by using camera and 3D laser scanner. Moreover, we present an optical flow based approach for segmenting similar motions.

1.3.2 Chapter 3

This chapter describes a trainable system for detection and tracking of pedestrians and cars. Here range data classification and image data classification is explained as well as the probabilistic fusion and tracking techniques.

1.3.3 Chapter 4

This chapter explains how to understand geometrical relations in detected objects in a scene by exploiting repetition patterns found in an image. This leads to an algorithm for probabilistically inferring occurrences of objects and for compressing data.

1.3.4 Chapter 5

The last chapter is dedicated to conclusions, discussions about each work and future works.

1.3.5 Appendix A

This appendix describes the robotic platform used in this thesis and it gives an overview of its sensor configuration.
1.4 Mathematical notations

In this section, we present the mathematical notations used in the following chapters.

- a \(d\)-dimensional vector will be notated with a bold letter, e.g. \(\mathbf{x} = (x_1, \ldots, x_d)^T \in \mathbb{R}^d\). If not otherwise noted, all vectors are considered as column vectors.

- a matrix with \(m\) rows and \(n\) columns will be denoted with a bold capital letter, for example

\[
A = \begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{pmatrix}
\]  (1.1)

The transpose of a matrix \(A\) will be denoted as \(A^T\).

- number constants will be notated with capital letters.

- all types of sets will be denoted with calligraphic capital letters. For example, a set containing \(N\) vectors \(x_1, \ldots, x_N\) will be denoted as

\[
\mathcal{X} = \{x_1, \ldots, x_N\}
\]  (1.2)

The cardinality of a set \(\mathcal{X}\) is expressed by the notation \(|\mathcal{X}|\).
Scene analysis plays an important role in many mobile robotics tasks, by introducing a cognitive aspect to the simple data collection process. Several robotics application including autonomous navigation, mapping, object detection and tracking could be enhanced by analyzing and understanding the scene of a complex environment, for example an outdoor scenario.

In the works presented in this chapter (published as Spinello and Siegwart [2007, 2008b]) we concentrate on that part of scene analysis that takes care of detecting and segmenting regions of interest for a mobile robot. The environment surrounding a robot, in urban or outdoor environments, contains a lot of complex information to process and to select. Our principal aim is to obtain a labeling of the environment by using a set of rules of interest for a particular application. An example could be to detect an artefact immerse in a natural outdoor environment (Spinello and Siegwart [2007]), in case of surveillance robotics, or to segment consistent moving objects (Spinello and Siegwart [2008b]), in case of autonomous tracking systems. Such regions of interest generators do not aim to be the solution for the wider problem of object detection or tracking, but they try to constrain and help further reasoning in selected areas of the environment.

### 2.1 Related works

Several methods exist in computer vision literature to define regions of interest in images. Interest point detectors and region detectors define salient image regions. The seminal work on scale invariant interest points and Gaussian pyramids Crowley and Parker [1984] fostered the development of several detectors: Shi and Tomasi detector (Shi and Tomasi
CHAPTER 2. MULTIMODAL ROI GENERATION

(Harris and Hessian affine (Mikolajczyk and Schmid [2002]), Laplacian of Gaussian (LoG) and Difference of Gaussians (DoG) (Lindeberg [1998]) are techniques based on local image gradient analysis to find local maxima representing corners or blobs; FAST (Rosten et al. [2009]) locates a stable region if a connected cluster of pixels has enough contrast in a circular neighborhood; SUSAN (Smith and Brady [1997]) computes an exponential function on a circular mask; Maximally Stable Extremal Regions (Matas et al. [2002]) detect stable image regions by using intensity thresholding. A comparative study showed the strength of these approaches (Mikolajczyk et al. [2005]). It is important to notice that these methods usually converge in regions where distinctive gradient change is present and/or where a space-scale stable blob is detected. No higher level reasoning is processed. Other concepts of region of interest generators have been also evaluated by Privitera and Stark [2000] inspired by human eye fixations. Other biologically inspired works, like the one of Heidemann [2004], exploit local color symmetries to segment peculiar image areas. Region of interest generation could be intended as a set of methods for obtaining object detection/recognition: literature related to this topic by using vision and range data is presented in Chapter 3.

In the field of range data processing, fewer temptatives have been made to generate regions of interest. In 2D laser range data, corners and lines are computed by using standard clustering algorithm like the ones compared by Nguyen et al. [2007a]. In 3D range data, local region descriptors are computed on sets of points. The most used are: Spin Images descriptors, introduced by Johnson [1997], that store the points of a specified area in a cylindrical histogram; Moment Grids descriptors (Zlot and Bosse [2008]) are a rectilinear voxelization of the space around a keypoint; Spherical Harmonic Descriptors, of Kazhdan et al. [2003], introduce a spherical harmonic representation to encode rotational invariant descriptors. These descriptors have been successfully used in several fields, like object classification (e.g. Triebel and Burgard [2007]) or 3D SLAM (Cole and Newman [2006]).

The topic of generating regions of interest for the specific task of detecting man-made structures in natural environments got relatively low attention in the field of robotics. Most of the methods in literatures are based on computer vision techniques. Few are the examples that use other sensors, mostly underwater sonars (Liu et al. [2006]), open sea radars (Ligthart et al. [2002]), peculiar lidar applications (Songxin et al. [2007]) and space remote sensing (Weed et al. [2002]). Kumar and Hebert [2003] propose, by using images only, a prior model based on multiscale Conditional Random Field to capture the local dependencies in the data by using a multiscale feature vector. Other works concentrate in detecting buildings (Krishnamachari and Chellappa [1996], Lin and Nevatia [1998], Mayer [1999]), a usual instance of man-made structures, in aerial imagery. The majority of these techniques use priors constituted by the detection of the roof in an image as the main cue for a building. These methods use grouping of low level image primitives (edges, lines) by employing heuristics rules or Markov Random Fields. Other works, in computer vision, concentrate in classifying the whole image as a natural landscape or urban scene by using principal components of the power spectrum (Oliva and Torralba [2001]) or by using edge coherence histograms (Anil et al. [1998]). Similar to our aim,
2.1. RELATED WORKS


Region segmentation in image sequences has the objective of segmenting areas with similar motion. Motion segmentation can be used to create a dynamic region of interest in which potential objects could be found. The hardest problem is to obtain a generalized method that is effective when the observer is not static and the observed objects are moving. An approach to motion segmentation is to assume that the motion present in an image sequence can be distinguished in several independent moving layers. The task is then to assign pixels to layers, to compute a motion parameterization and to determine the number of total layers. Expectation-Maximization (EM) is a well known technique that has been successfully applied to this problem (Ayer and Sawhney [1995], Jojic and Frey [2001], Wang and Adelson [2004], Y. Weiss and Adelson [1996]). Graph cuts have been also employed as another solution for the same optimization (Wills et al. [2003], Xiao and Shah [2005]). Another way of approaching the problem is to formulate the optical flow segmentation as a multi-body factorization, which is solved by determining independent subspaces and setting constraints over a number of frames (Ke and Kanade [2001], Machline et al. [2002]). Other approaches cluster the motion models of pixels by using eigenvector analysis (Shi and Malik [1998]) or minimize a functional over a spatio-temporal volume using spatio-temporal gradients (Cremers and Soatto [2005]). Rothganger et al. [2004] group the sequence in locally coherent regions by constraining feature correspondence. Tong et al. [2004] use a tensor voting approach to obtain geometrical estimation and motion segmentation. Vidal [2005] segments motion subspaces in closed form by using Generalized Principal Component Analysis.

In this chapter we address region of interest generation as a way of segmenting the environment in regions where artefacts could be found and a way of clustering similar motions in a sequence of images. None of the vision based or range based regions of interest generators/descriptors here discussed are well suited for detecting man-made structures in natural environments. A specially tailored measure of likelihood needs to be formulated for this specific task. We address this problem by using properties of range data smoothness and color uniformity that are distinctive cues in an unordered outdoor natural scenario. Moreover, no other work utilizes a multimodal approach that could overcome the limitations associated to single-sensor setups; we address this problem by using a vision and 3D range data probabilistic fusion approach. Most of the reviewed techniques that achieve motion segmentation in image sequences require several image frames to produce a solution or require of heavy computational resources. Instead, we tackled this problem by providing a two-frames only approach based on a novel measure of local optical flow similarity and we formulated the problem as a probabilistic clustering task.
CHAPTER 2. MULTIMODAL ROI GENERATION

2.2 Unsupervised Segmentation of Artefacts in Natural Outdoor Environments

Highly structured environments, like indoor scenarios or office spaces, are mainly composed of elements with simple geometrical structures: walls, cupboards, cabinets and tables have usually flat surfaces. Thus, the environment could be often modeled as a set of orthogonally arranged primitives (lines and planes). Some algorithms in robotics, like indoor Simultaneous Localization and Mapping (SLAM) (Nguyen et al. [2007b]), use this prior knowledge to improve the performance and the quality of the results. In contrast, the method explained here exploits the unstructured nature of outdoor environments to segment areas in which artefacts could be found. We propose a multimodal method based on an unsupervised segmentation technique of 3D range data and image data. We define an area in which artificial objects could be present in outdoor environments using structure and appearance information: an artificial object is composed of a collection of smooth surfaces with sufficiently extended area and distinctive colors with respect to the environment. These assumptions try to generalize a wide number of man-made objects. Range data processing is computed for obtaining structure information by a segmentation of 3D smooth surfaces; appearance information is obtained from image processing by clustering distinctive colors. Nevertheless, it is possible to have cases in which structural information is more relevant than appearance information (or viceversa). It is therefore necessary to manage the fusion of the information in a probabilistic manner.

Structure and appearance information have to be merged together in a common space. Each point of each segmented 3D surface and color blob is probabilistically fused using a Bayesian modeling approach. The resulting map is then clustered by using a 2.5D Self Organizing Network in order to label and segment areas.

2.2.1 Contributions

The novelties introduced are as follows:

- A lightweight unsupervised method, conceived to be fast and deployable for field robots, to segment the space into regions of interest in order to generate hypotheses for further reasoning (e.g. object classification).

- The extraction of relevant features from 3D data and images to detect man-made artefacts in a natural outdoor environment.

- A probabilistic fusion method of 3D range data and camera images on a common fusion space by using Bayesian reasoning and 2.5D Self Organizing Networks (SON).

2.2.2 Structure information processing

Salient structure information for artificial objects in outdoor environments is defined by a metric on the smoothness of surfaces from the 3D range dataset. An outdoor
2.2. SEGMENTATION OF ARTEFACTS IN NATURAL ENVIRONMENTS

scenario is a highly unstructured environment: natural objects are generally constituted by irregular 3D shapes with complex surface profiles (e.g. trees, leaves, rocks etc). Instead, human-made objects are usually symmetrical and have smooth surfaces, due to common design/object manufacturing processes. It is therefore acceptable, for the aim of this work, to consider certain surface properties, like smoothness and area extension, interesting geometrical cues of an artificial object.

2.2.2.1 Local plane fitting

In order to define regions of common smoothness a process to compute local tangent planes is needed. Given a point \( x_i = (x_i, y_i, z_i) \), from the 3D range data set, it is possible to define its \( k \)-neighborhood \( N^k_{x_i} \) as the \( k \)-closest points that satisfy the distance criterion \( \| x_i - x_j \| < M \). The fitted plane in the neighborhood is defined by the centroid \( o_i \) and the normal \( n_i \).

The centroid \( o_i \) of \( N^k_{x_i} \) is computed by calculating the geometrical center of the \( k \)-neighborhood subset:

\[
o_i = \frac{\sum_{x_j \in N^k_{x_i}} x_j}{k}
\]

The normal \( n_i \) of \( N^k_{x_i} \) is obtained by using the principal component analysis method proposed by Hoppe et al. [1992]. In this case, the covariance matrix \( C_i \) associated to \( N^k_{x_i} \) is a symmetric \( 3 \times 3 \) positive semi-definite matrix, composed by:

\[
C_i = \sum_{x_j \in N^k_{x_i}} \left( (x_j - o_i) \otimes (x_j - o_i) \right)
\]

where the symbol \( \otimes \) denotes the outer product operator. The eigenvalues of the matrix \( C_i \) denoted as \( \lambda^1_i \leq \lambda^2_i \leq \lambda^3_i \) are associated with unit eigenvectors \( v^1_i, v^2_i, v^3_i \) and the corresponding normal vector of the fitted plane is \( n_i = \pm v^1_i \). Each local normal has \( \|n_i\| = 1 \). The sign determines the orientation of the plane and it has to be consistent with the nearby planes. This problem is addressed using the knowledge that the normals of the planes can be orientated only in the opposite direction of the scanning laser beam.

The process of plane fitting is iterated for every point of the data set. Therefore, each point is associated to a \( k \)-neighborhood subset, a centroid and a normal vector defining a local orientation.

2.2.2.2 Smooth surfaces extraction

After the local plane fitting step, further geometrical processing is needed to segment smooth regions. A region growing approach is run: the normal orientation associated to each centroid is compared with orientations of the neighboring points by using a special ordering. Points are triangulated between each two consecutive line-scans, thus forming a searching order for each point. When two neighboring normals have an angle \( \alpha_i \) smaller than the threshold \( \alpha_{max} \), then the region is grown with that point. A new region is created
CHAPTER 2. MULTIMODAL ROI GENERATION

when no neighboring normal has an angle $< \alpha_{\text{max}}$. It is possible to describe this process more in detail by following several steps:

1. Choose the first not visited centroid and consider its normal as Representative Vector $\mathbf{n}_r$. The region $\mathcal{R}_i$ is created.

2. Select the following not visited neighboring centroid and consider its normal as Candidate Vector $\mathbf{n}_c$.

3. Compute, using the scalar product, $\alpha = \cos(\mathbf{n}_r \cdot \mathbf{n}_c)$, if the condition $\alpha < \alpha_{\text{max}}$ is satisfied then $\mathbf{n}_c$ is added to region $\mathcal{R}_i$, otherwise return to step 2.

4. The new Representative Vector is computed as $\mathbf{n}_r = \frac{\sum_{\mathbf{n}_j \in \mathcal{R}_i} \mathbf{n}_j}{\|\mathcal{R}_i\|}$ and then it is normalized. The centroid is marked as visited. Return to step 2 or go to step 1 until all points are visited.

Step 4 is calculated to obtain a good surface following during the region growing. The parameter $\alpha_{\text{max}}$ is chosen beforehand. It expresses the tolerance among regions orientation. When every normal has been analyzed, points are grouped in smooth regions. A value is stored for each region (that contains at least 3 points) associated with the inverse area measurement:

$$ h^l_i = 1 - \frac{1}{\|\mathcal{R}_i\|} \quad (2.3) $$

The set $\mathcal{H}^l = \{h^l_1, h^l_2, \ldots h^l_w\}$, where $h^l_i \in [0, 1]$, defines the values associated to the $w$ smooth surface patches extracted from the 3D range data.

Smooth horizontal surfaces extracted from this processing step are managed with special care. Their value $h^l_i$ is weighted by using a penalty value. This value is proportional to the position of the surface with respect to the $Z$ axis: $h^{\text{hor}}_i = \bar{\mathcal{R}}^Z_i$, where $\bar{\mathcal{R}}^Z_i$ represents the average $Z$ value of the horizontal surface $\mathcal{R}_i$ in the laser coordinate system. Intuitively, we assume that a smooth horizontal surface found at a high $Z$ position does not correspond to a part of the ground plane, and vice versa:

$$ h_i^l = h_i^l h_i^{\text{hor}} \quad (2.4) $$

Thus, a low weight value $h_i^l$ is assigned to the ground plane thanks to equation (2.4).

Furthermore, an adjacency graph is built among the regions to represent the geometric topology. Each node represents a region, each arc connects two regions if they are adjacent. A graph exploration algorithm is thus applied to link contiguous smooth surfaces by using a connected component algorithm (Cormen et al. [1990]).

2.2.3 Appearance information processing

The purpose of image processing in this work is to obtain the appearance information of areas where artefacts could be found. In order to define salient color blobs in the picture a two-steps technique is applied: color clustering and color quantization.
2.2. SEGMENTATION OF ARTEFACTS IN NATURAL ENVIRONMENTS

Before proceeding, two morphological image operations are used to simplify color diversity in the image: erosion followed by a dilation. The color space selected for the image processing is the HSB (Hue, Saturation and Brightness) due to the similarities of this color model to the way humans tend to perceive colors. Image dilation and image erosion consider each channel of an HSB image as a grayscale array. Dilation computes, for each image pixel, the maximum value of its neighboring pixels. Erosion takes, for each image pixel, the minimum value of its neighboring pixels. The neighborhood has been defined by a circle of 5 pixels radius. The morphological operation of erosion and dilation is called opening (Serra [1983]). Figure 2.1 shows the effects of this morphological opening on a color image.

2.2.3.1 Color Clustering

The idea of this processing is to remove the background color of the image, that is assumed to be the most present in the picture, in order to retain only the colors interesting for the appearance analysis of the scene. A simple color selection is not enough, because we want to be robust to small hue and color saturation change. Therefore, we need to select, in an unsupervised manner, which is the group of colors that represent the background. The purpose of any clustering method is to group entities on the basis of similarity of features. In our case we need to segment, in the color space, the main cluster to detect the principal color present in the image.

We feed the hue and saturation image data to a Self Organizing Network (SON) for clustering. A SON is a neural network that has been used for obtaining unsupervised learning and clustering with low computational demand, as shown in the work of Vasquez and Fraichard [2006]. This method has the advantage of adaptively computing the number of clusters with a low computational complexity. For this task, we employ an
SON with a 2D squared grid topology. The main color cluster is detected by calculating the biggest cluster area in the color space.

### 2.2.3.2 Color Quantization

After removing the background color, we aim to exclude a palette of ‘outdoor’ natural colors. The coarse filtering of discarding green, brown, and too dark colors is achieved by using several windows of non-admissible color in the HSB image color space. This process introduces some a-priori knowledge in the algorithm, but it represents a reasonable assumption for natural scenarios. Outdoor environmental brightness varies during the day, therefore we need to design a color removal process that is adaptive to the environment brightness changes. Thus, we computed $B_{\text{env}}$ as the median environmental brightness and we set a minimum brightness threshold for all the coarse color filtering windows to $B_{\text{min}} = \frac{B_{\text{env}}}{2}$.

The Sobel operator (Sobel and Feldman [1968]) is applied on each channel of each color cluster to compute an approximation of the gradient of the image intensity. Therefore the sets $B^H_i, B^S_i, B^B_i$ relative to the extracted $i$-surface contain the values of the gradient for each channel. Using the information theory, it is possible to compute the blob entropy value (Gonzalez and Woods [1993]), considering each channel as an independent intensity image:

$$
\begin{align*}
S^H_i &= \frac{\text{hist}(B^H_i)}{\|B^H_i\|}, \\
S^S_i &= \frac{\text{hist}(B^S_i)}{\|B^S_i\|}, \\
S^B_i &= \frac{\text{hist}(B^B_i)}{\|B^B_i\|}
\end{align*}
$$

where \(\text{hist}\) is a function that computes the histogram (with a defined quantity of bins $n$) of the gradient distribution in a set.

$$
\begin{align*}
h^c_i &= \left[- \sum_n g^H_i \log_2(g^H_i)\right] + \left[- \sum_n g^S_i \log_2(g^S_i)\right] + \left[- \sum_n g^B_i \log_2(g^B_i)\right]
\end{align*}
$$

The set $H^c = \left(1 - \frac{h^c}{h_{\text{max}}}, 1 - \frac{h^c}{h_{\text{max}}} \ldots , 1 - \frac{h^c}{h_{\text{max}}} \right)$, where $h_{\text{max}} = \arg\max_{[1, a]}(h^c)$, defines the values associated to the set of $u$ extracted color blobs. Image entropy is a quantity which is used to describe the amount of information richness present in the image blob. The lower the entropy associated to the color patch, the higher the confidence of being a part of an artefact. A low energy patch is a more reasonable evidence of an artificial object as it encodes an almost uniform color patch.

### 2.2.4 Information Fusion

Structure and appearance characteristics of artificial objects have been computed. Therefore a careful merging of these incomplete information it is necessary to handle the partial and noisy information.

Structure information is defined in $\mathbb{R}^3$ space, appearance information in $\mathbb{R}^2$ space. A dimensionality reduction is defined for the structure information space from $\mathbb{R}^3 \rightarrow \mathbb{R}^2$.
by projecting the selected smooth surfaces in the image (after a calibration of the camera-laser system). Furthermore, the structure information that falls out of the boundaries of the viewing area is discarded.

Structure and appearance information are considered cues of same importance, thus the same confidence level of detection should be given in the fusion information process. The information fusion is addressed using a Bayesian modeling approach. The variables used to formalize the problem are:

- \( \phi^{x,y} \): it is a binary variable that describes the existence of an artificial object.
- \( \theta^{l,x,y} \): it encodes the structure information given by the laser clustered points.
- \( \theta^{c,x,y} \): it encodes the appearance information expressed by the image color blobs.

The notation \((\cdot)^{x,y}\) in the previous formulas denotes that the variable is computed in the position with coordinates \((x, y)\) in the fusion plane.

Starting from the joint distribution and applying recursively the conjunction rule we obtain the decomposition:

\[
p(\phi^{x,y}, \theta^{l,x,y}, \theta^{c,x,y}) = p(\phi^{x,y}) \cdot p(\theta^{l,x,y} | \phi^{x,y}) \cdot p(\theta^{c,x,y} | \phi^{x,y})
\]

In equation 2.7 the phenomenon \(\phi^{x,y}\) is considered to be the main reason for the contingency of the structure and appearance information, thus knowing the cause \(\phi^{x,y}\) of the readings the variables \(\theta^{l,x,y}\) and \(\theta^{c,x,y}\) are independent. In general, this hypothesis is not always satisfied, but it is often used in literature and it has the main advantage of considerably reducing the complexity of the computation.

The conditional probability that defines the information fusion is:

\[
p(\phi^{x,y} = \text{true} | \theta^{l,x,y}, \theta^{c,x,y}) = \frac{p(\phi^{x,y} = \text{true}) \cdot p(\theta^{l,x,y} | \phi^{x,y} = \text{true}) \cdot p(\theta^{c,x,y} | \phi^{x,y} = \text{true})}{\sum_{\phi^{x,y} = \text{true, false}} (p(\phi^{x,y}) \cdot p(\theta^{l,x,y} | \phi^{x,y}) \cdot p(\theta^{c,x,y} | \phi^{x,y}))}
\]

The equation (2.8) is evaluated for each point of the fusion plane. \(p(\theta^{l,x,y} | \phi^{x,y} = \text{true})\) is defined for each point by the value \(h^l_i\) of the associated surface patch. \(p(\theta^{c,x,y} | \phi^{x,y} = \text{true})\) is defined for each point by the value \(h^c_j\) of the relative color blob. If the method of artificial object detection is, for example, used as a part of a warning system, then false positive rates can be tolerated and \(p_s\) in equation (2.9) can be set to a high value. In contrast, if it is included as part of active vehicle control a more conservative choice is needed, see Figure 2.2.

The resulting probability in the fusion plane has to be clustered to segment distinct areas. The idea is to obtain a segmentation that is sensitive to the probability values of each point. Thus, we proposed an update to the theory of the Self Organizing Network in order to locally segment probabilistic data by considering 2.5D values, see Section 2.2.4.1.

After clustering, a label and an average probability value are finally assigned to each detected cluster. The fusion plane is therefore completely segmented in regions where, with certain confidence, artificial objects could be found.
Figure 2.2: Effects of the prior \( p(\phi^{x,y} = \text{true}) \) on the probabilistic fusion process. On the axes \( X-Y \) are shown the structure and appearance likelihood expressed by \( p(\theta^{x,y}_i | \phi^{x,y} = \text{true}) \) and \( p(\theta^{x,y}_c | \phi^{x,y} = \text{true}) \). The Z value expresses the resulting fused probability of equation (2.8). On the left figure the prior is set to 0.2, on the right figure it is set to 0.8. From these figures we notice that by increasing the prior, the resulting confidence on the computed likelihood data increases, thus the fused probability increases. The higher the prior, the softer the fusion between the two values.

2.2.4.1 2.5D Self Organizing Network for clustering

The 2.5D Self Organizing Network clustering method here developed has been mostly inspired by the work of Vasquez and Fraichard [2006]. A SON is composed by a network of \( T = M \times N \) nodes connected each other with undirected edges arranged in a circular or squared grid, with \( M \) rows and \( N \) columns. Every node is connected with four other nodes (excluding the nodes located on the network border) and it has two associated variables: its position \( \mu_i = (\hat{x}_i, \hat{y}_i) \) and a value \( c_i \). For every arc that connects the node \( i \) with the node \( j \) the weight \( e_{i,j} \) is stored. We here introduce a circular topology SON, that has been used in the experiments for segmenting the fusion plane. The SON is initialized by computing the Cartesian coordinates of each node and by setting the network parameters:

\[
\begin{align*}
\hat{x}_i &= (R_m + m \cdot R_\Delta) \cos(\alpha + n \cdot \alpha_\Delta) & (2.10) \\
\hat{y}_i &= (R_m + m \cdot R_\Delta) \sin(\alpha + n \cdot \alpha_\Delta) & (2.11) \\
c_i &= 0 \ \forall_i & (2.12) \\
e_{i,j} &= 0 \ \forall_{i,j} & (2.13)
\end{align*}
\]
where $R_m$ is the minimum radius of the SON grid, $R_\Delta$ is the radius increment, $\alpha$ is the current angle, $\alpha_\Delta$ is the angle increment, $m \in [0, M] \in \mathbb{N}$ and $n \in [0, N] \in \mathbb{N}$. $R_\Delta$ and $\alpha_\Delta$ are chosen beforehand and they define the node quantity in each circle and radius.

During the initialization of the algorithm, every node is placed to obtain a regular circular grid, see Figure 2.3-left, and every $c_i$ and $e_{ij}$ are set to zero. The aim is to adapt the network to the data, by harmoniously moving the nodes accordingly to the position and the weight of each data point. This adaptation is generally mentioned as learning and it is processed every time an input data point is fed to the SON.

The vector $x = (x, y, v)$ defines a data point expressed by the Cartesian coordinates $(x, y)$ and the weight $v \in [0, 1]$. For notation clarity we denote $x_{xy}$ as the vector consisting in only the Cartesian coordinates of the data input $x$ and $x_v$ consisting in only the value $v$ associated to the vector $x$. The first step of the learning phase consists in the selection of the two network nodes closest to the input data. Only the Cartesian coordinates of the data input $x$ are considered in this process:

\[
w_1 = \arg \min_{i \in [0, T]} \| x_{xy} - \mu_i \| \tag{2.14}
\]
\[
w_2 = \arg \min_{i \in [0, T] \setminus w_1} \| x_{xy} - \mu_i \| \tag{2.15}
\]

where $x_{xy}$ represents the vector consisting in only the Cartesian coordinates of the data input $x$. Then, the counter of the winning node $c_{w_1}$, the counters of the neighboring nodes $c_i \forall i \in N_{w_1}$ and edge weight $e_{w_1,w_2}$ are updated:

\[
e_{w_1,w_2} \leftarrow e_{w_1,w_2} + 1 \tag{2.16}
\]
\[
c_{w_1} \leftarrow 1 + c_{w_1} + x_v \tag{2.17}
\]
\[
c_i \leftarrow 1 + c_i + x_v \forall i \in N_{w_1} \tag{2.18}
\]

In equation (2.18) the counter dynamic $c_{w_1}, c_i$ is changed according to the weight value present in $x$. $c_{w_1}, c_i$ have been saturated to the maximum values $C_w$ and $C_i$. The mean of the closest node and its four neighbors is then modified, as shown in Figure 2.3-middle:

\[
\mu_{w_1} \leftarrow \mu_{w_1} + \frac{e_{w_1}}{c_{w_1}} (x_{xy} - \mu_{w_1}) \cdot x_v \tag{2.19}
\]
\[
\mu_i \leftarrow \mu_i + \frac{e_i}{c_i} (x_{xy} - \mu_i) \cdot x_v \forall i \in N_{w_1} \tag{2.20}
\]

where the set $N_{w_1}$ contains the indices of the neighboring nodes of node $w_1$, the parameters $0 < e_i < e_{w_1} < 1$ regulate the network adaptation speed.

The update equations (2.18), (2.19) and (2.20) are modified with respect to the theory proposed in Vasquez and Fraichard [2006] to keep in account the third coordinate that define the 2.5D space. It is important to notice that the mean $\mu_i$ moves more if a high value $x_v$ is present in the input data. Viceversa, if $x_v$ is a small value, $\mu_i$ moves just a little. The learning phase stops when all the data is fed to the SON. Then, the cluster representation is accomplished by using a graph cutting approach: arcs having low weight values $e_{ij}$ are cut, leaving connected components with high edge values (see Figure 2.3-right). The
idea is that high values of $e_{ij}$ correspond to a high likelihood that nodes $i$ and $j$ belong to the same object.

This update to the clustering method, used in this work for segmenting the information fusion, has several advantages:

- The maximum cardinality of detectable clusters is not defined by the user.
- The algorithm complexity of the algorithm is $O(T \cdot D)$, where $D$ is the number of input points.
- The clusters shapes are influenced by the weight value present in each point of the map.

2.2.5 Experimental results

We demonstrate the performance of the proposed algorithm using a data set acquired in a real outdoor environment to test the detection of artificial objects.

The platform used to acquire data is composed of a rotating laser rangefinder and an omnidirectional camera mounted on a movable tripod. An omnidirectional camera is a camera with a 360-degree field of view in the horizontal plane. With this setup we are able to capture a full panorama of the environment with both sensors.

The laser rangefinder is mounted with its scan line in vertical position; the rotation in the $z$ axis is given from a step motor. The revolution frequency of the laser is 0.5Hz. In a complete rotation the laser scans $[-50^\circ, 50^\circ]$ in the elevation plane and $[0^\circ, 360^\circ]$ in the azimuthal plane. The laser sensor has an elevation angle resolution of 1°. The omnidirectional camera is constituted by a standard firewire camera with an hyperbolic lens. The images captured have the resolution of $640 \times 480px$. The omnidirectional camera is calibrated by using the method proposed by Scaramuzza et al. [2006] with the 3D laser. Thus, the mathematical function that defines the correspondence between laser

Figure 2.3: **Left:** Lattice of a circular SON. **Middle:** Data is inserted and nodes coordinates are modified. **Right:** Clusters are defined via graph cutting.
data points and image data points is achieved. Moreover, the origin of the laser reference frame has been set in the contact point of the tripod with the ground.

The data retrieved from the 3D rotating laser is processed to compute local normals, as presented in Figure 2.4. Smooth linked surfaces are identified and are shown in Figure 2.5. Note that the radial sectors missing in Figure 2.4 and Figure 2.5 are not caused by the algorithm of range data processing but they are caused by lack of data retrieved from the 3D laser. The data is then retrieved from the omnidirectional camera. The sky is removed by a circular area of interest: the minimum bounding circle is set beforehand and represents the minimum useful radius of the field of view. The most distant point from laser range data set is taken, then its 2D correspondence in the image plane is computed and the maximum radius for the image area of interest is set. The removal of the main color region is presented in Figure 2.6-left. Therefore the color quantization is executed and the final image processing results are shown in Figure 2.6-right.

The 3D smooth surfaces from the laser data processing are projected onto the omnidirectional camera plane and the information fusion takes place, see Figure 2.2.5-left. The 2.5D self organizing network used is a neural network with circular topology in order to obtain a better data coverage in the fusion plane, which is the omnidirectional camera image plane. Therefore, the segmentation is performed, see Figure 2.2.5-right.

Three men, some cars, and some colored boxes are detected in the scene, every cluster
Figure 2.5: The surfaces extracted from a 3D range data set are depicted in color.

Figure 2.6: Omnidirectional camera ROI is defined by the two red circles. Results of color clustering are shown in the left figure, final color blobs in the right figure.

with probability $> 0.25$ is shown in the 3D model figure presented in Figure 2.8. As a quantitative result, data has been collected and labeled for 8 different snapshots in an outdoor environment. 63.2% of labeled areas in which artefacts are present have been successfully recognized.

Even though the image processing involves a dynamic brightness technique, the algorithm still suffers from poor light conditions. In such case the resulting object detection performance decreases because, in the fusion step, extracted color patches are represented
2.2. SEGMENTATION OF ARTEFACTS IN NATURAL ENVIRONMENTS

Figure 2.7: **Left**: Fusion plane is populated with structure information and appearance information probability. **Right**: Clusters expressing artificial objects are segmented from the fusion plane. The colorbar describes the clusters confidence.

by small islands of low contrast that obtain low likelihoods. Similar poor results are obtained if materials with low reflectivity are present in the scene. In that case the fusion process will be affected by the poor surface extraction process that would produce empty areas or wrongly estimated surface normals. The distance also plays an important role in the detection phase. Distant surfaces are low detailed due to the point scarcity. In the

Figure 2.8: 3D point cloud in which big colored points represent detected artificial objects.
experimental platform used, the hyperbolic mirror of the omnidirectional camera gives a very small resolution of the peripheral image (that part of the image that is far from the center), resulting in a poor detection of far away man-made objects.

To optimize the performance, the 3D laser and image processing are computed in parallel threads. The algorithm is completely implemented in C code and it takes 1.0s to generate the results on a Intel Centrino 1.60 Ghz based system.
2.3 Motion Segmentation Using a Local Entropy Field

Region of interest (ROI) generation in image sequences has a considerable importance in a number of mobile robotics applications using vision, especially in object detection and tracking in dynamic and outdoor environments. It potentially reduces computation time by focusing in important areas of the image. The aim of the work explained in this section is to segment regions of interest that contain independent relative motions in the optical flow and to use these moving regions to further enhance the segmentation in the following frames. The technique proposed generates from two frames a motion segmentation and it is designed to work in environments with moving observers and moving objects. This kind of segmentation is particularly interesting for mobile robotics where time might be critical, specially if this method is associated to a navigation module in dynamic environments or to a surveillance task.

The algorithm has been built considering a practical implementation with compact memory usage, suitable for mobile robotics. An schematic overview of this work is given in Figure 2.9.

2.3.1 Contributions

The novelties of this work are as follows:

- the definition of a lattice in the motion vector field through the use of a Delaunay triangulation.
- the generation of a local entropy field as a description for local motion diversity in the optical flow.
- the usage of a Markov Random Field (MRF) for defining similar motion boundaries by using the information of local entropy values.
- The usage of a motion prediction scheme for enhancing motion segmentation of future image frames.

2.3.2 Sparse Optical Flow

The essential element of motion segmentation is a method to compute optical flow from a sequence of images. A sparse optical flow technique has been chosen: only points in the image with high information content are used as reference to reliably compute the image velocity. While these methods are potentially faster and more reliable than dense optical flow methods, they have the drawback of obtaining a velocity information in smaller number of points in the image. A feature tracker based on the work of Tomasi and Kanade [1991] is here used.

The Kanade-Lucas-Tomasi features tracker (KLT) is based on the assumption that images taken at near time instants are usually strongly related to each other, because they refer to the same scene taken from only slightly different viewpoints. Formally, this
means that an image sequence, function of the variables space \(px, py\) and time \(t\), \(I(px, py, t)\) is not arbitrary, but satisfies the following property:

\[
I(px, py, t + \tau) = I(px - \xi, py - \eta, t) \quad (2.21)
\]

This means that a later image taken at time \(t + \tau\) can be obtained by moving every point in the current image, taken at time \(t\), by a suitable amount. The amount of motion \(z = (\xi, \eta)\) is called the displacement of the point \(px = (px, py)\) between time instants \(t\) and \(t + \tau\), and it is in general a function of \(px, py, t\) and \(\tau\). The equation (2.21) is by large satisfied on objects surface points, far from occluding contours. We define \(J(px) = I(px, py, t + \tau)\), and \(I(px - z) = I(px - \xi, y - \eta, t)\), where the time variable has been dropped for brevity. Our local image model is defined as: \(J(px) = I(px - z) + n(px)\) where \(n\) represents noise. The displacement vector \(z\) is then chosen in order to minimize the residual error defined by the following double integral over the given image area window \(W\):

\[
\epsilon = \int_W \left[ I(px - z) - J(px) \right]^2 F \, dp_x \quad (2.22)
\]

In this expression, \(F\) is a weighting function that is commonly set to 1. The vector \(z\) defines an optical flow vector.

### 2.3.3 Features Topology

In the sparse optical flow, the velocity vectors are scattered in the image plane. The neighborhood of a feature is defined through the use of a Delaunay triangulation (Delaunay [1934]). The origins of the velocity vectors of the computed optical flow are used as the set of points \(P\) that define the vertices of the triangulation. Any point of \(P\) is inside the circumcircle of any triangle in the triangulation. Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the set. The edges of the graph defined by the Delaunay triangulation define the search path among features. Delaunay triangulation has been selected for its low computational complexity (in the average case: \(O(||P||)\)) and its unique subdivision of the space.
2.3. MOTION SEGMENTATION USING A LOCAL ENTROPY FIELD

2.3.4 Local entropies field

We address the problem of optical flow sparsity by tessellating the image frame in cells. The dual of the Delaunay triangulation, the Voronoi decomposition, adaptively tessellates the image frame in cells. Each cell, a polygon with an area of several pixels, is assumed to have a uniform local optical flow motion. We use this cell decomposition to produce an informative descriptor of local motion diversities: a local entropy potential. The aim is to encode, in a compact form, the amount of motion diversities between a considered cell and its adjacent neighborhood: the difference between motion vector orientations, magnitudes and features distances. It is possible to formulate the entropy of a discrete random variable \( b \) that has possible outcomes \( (b_1, b_2, \ldots, b_n) \) as

\[
\varepsilon = \sum_{i}^{n} p(b_i) \log_2 (p(b_i))
\]

where \( p \) denotes the probability mass function of \( b \). In case that \( p(b_i) = 0 \), the value of \( 0 \log_2 (0) \) is taken to be 0 consistently to the limit \( \lim_{p \to 0^+} p \log p = 0 \).

We produce for each cell \( i \), containing an optical flow vector \( z_i \), three sets of differences by considering the set adjacent cells \( K = (k_1, \ldots, k_f) \) in the Voronoi tessellation:

- a set of differences in the local optical flow vector orientation:
  \[ A = \left( \arccos \left( \frac{z_i \cdot z_{k_1}}{\|z_i\|\|z_{k_1}\|} \right), \ldots, \arccos \left( \frac{z_i \cdot z_{k_f}}{\|z_i\|\|z_{k_f}\|} \right) \right) \]

- a set of differences in the local optical flow vector magnitude:
  \[ M = (\|z_i\| - \|z_{k_1}\|, \ldots, \|z_i\| - \|z_{k_f}\|) \]

- a set of distance between vectors origin:
  \[ D = (\|z_i - z_{k_1}\|, \ldots, \|z_i - z_{k_f}\|) \]

From these sets we build three histograms \( H_A, H_D, H_M \), respectively composed of \( N_A, N_D, N_M \) bins. In the context of this work we introduced three vectors \( w^A, w^M, w^D \), one related to each histogram, that weight each histogram bin by considering more important the bins that express big optical flow differences:

\[
w^A = (w^A_1, \ldots, w^A_{N_A}) \quad w^M = (w^M_1, \ldots, w^M_{N_M}) \quad w^D = (w^D_1, \ldots, w^D_{N_D})
\]

To clarify: we define \( J^A, J^D, J^M \) as the sets that contain the indices of the elements associated to the bins \( q^A, q^D, q^M \) of the histograms of differences relative to the cell \( i \). The bin weights are then specified by:

\[
w^A_q = \sum_{g=1}^{\|J^A\|} \frac{a^A_g}{\pi} \quad w^D_g = \sum_{g=1}^{\|J^D\|} \frac{d^D_g}{l_{\text{max}}} \quad w^M_g = \sum_{g=1}^{\|J^M\|} \frac{m^M_g}{m_{\text{max}}}
\]

where \( a^A_g, d^D_g, m^M_g \) represent the indices of the \( g \)-element of the sets \( J^A, J^D, J^M \); \( a_i \subset A, d_i \subset D, m_i \subset M \). The values \( d_{\text{max}}, m_{\text{max}} \) have been set beforehand as the maximum values.
between two frames. An example of the effects caused by these weights to an histogram is given in Figure 2.10.

Thus, after each histogram is normalized, it is possible to compute the value $h_i$ associated for each cell as:

$$h_i = - \sum_{g=1}^{n_A} w_A^g p(H_A(g)) \log_2(p(H_A(g))) - \sum_{g=1}^{n_D} w_D^g p(H_D(g)) \log_2(p(H_D(g))) - \sum_{g=1}^{n_M} w_M^g p(H_M(g)) \log_2(p(H_M(g)))$$  \hspace{1cm} (2.25)

It is important to highlight that the local cell entropy expressed by 2.25 make the entropy value dependent on the mean of the histograms by using the weight vectors. In this way, histogram bins expressing big motion differences in the neighborhood play a bigger role in the computation of the local cell entropy. Note also that the equation expresses a semi-definite positive function.

![Figure 2.10: Two histograms $H_A$ are plotted on the left (in red and blue). The bins of the same two histograms are weighted with bin weights $w_A^g$ on the right figure. It is important to notice that the higher the angle difference, the higher the weight associated for each bin.](image)

The cell local entropy value expresses the weighted amount of information present in its neighborhood, using the cell as reference value. In general we can think that small values of $h_i$ highlight a local area of similar motion. Motion boundaries could be found in cells in which $h_i$ is high. An example of a computed local entropy field is shown in Figure 2.11.

### 2.3.5 Detecting Motion Boundaries with Markov Random Fields

We address the problem of detecting motion boundaries by modeling the problem with a pairwise Markov Random Field (MRF).
A Markov random field, is a graphical model in which a set of random variables have a Markov property described by an undirected graph. We define a vector $l = (l_1, l_2, \ldots, l_N)$ of $N$ random variables and $Y = (y_1, y_2, \ldots, y_N)$ as a set of observations of the hidden states $l$. We assume that there is some statistical dependency between $l_i$ and $y_i$, which we write as a joint compatibility function $\phi_i(l_i, y_i)$, or evidence of $l_i$. We encode the statistical dependency between random variables in $l$ by defining them as nodes of an undirected graph $G = (l, E)$, of $E$ edges. It is possible to represent a dependency between $l_i$ and $l_j$ with an arc between these nodes quantified by a compatibility function $\psi_{ij}(l_i, l_j)$. We can then specify the overall joint probability of the Markov Random Field as:

$$p(l, Y) = \frac{1}{Z} \prod_{ij} \psi_{ij}(l_i, l_j) \prod_i \phi_i(l_i, y_i)$$  \hspace{1cm} (2.26)

where $Z$ is a normalization constant. The first product in equation (2.26) is over all neighboring pairs of nodes, $i$ and $j$. In our case each hidden states $l_i$ could take the value of:

$$l_i = \{1, -1\}$$  \hspace{1cm} (2.27)

where the label $-1$ indicates a cell being a part of cluster boundary and the label $1$ a cell included in a cluster.

The arcs of the MRF lattice are defined by the Delaunay triangulation among the origins of optical flow vectors (see Section 2.3.3). A visual explanation of this structure is given in Figure 2.12-left: the circles depict nodes and the lines indicate the arcs of the graph.

The observations $y_i$ of equation (2.26) relate to the observed local entropy value $h_i$ explained in the previous section. The evidence function $\phi_i(l_i, y_i)$ makes use of a
preclassification stage and it is given by:

$$\phi_i(l_i, y) = e^{(\gamma - h_i) l_i} \quad (2.28)$$

where $\gamma \in \mathbb{R}$ represents a dividing hyperplane that has been set by experimentally tuning the desired sensitivity of local entropy potentials for boundary detection. Values on the left of the hyperplane express the class 1, values bigger than $\gamma$ are classified as $-1$. The equation (2.28) produces a small value if the hidden state is in disagreement with the preclassification state and viceversa. The compatibility function $\psi_{ij}(l_i, l_j)$ is defined as:

$$\psi_{ij}(l_i, l_j) = e^{(\gamma - h_i + h_j) g(l_i, l_j)} \quad (2.29)$$

$$g(l_i, l_j) = \begin{cases} 1 & \text{if } l_i = l_j = 1 \\ -1 & \text{if } l_i = l_j = -1 \end{cases} \quad (2.30)$$

The equation (2.29) returns high values if adjacent values are classified with the same class by their local entropy level. The exponential function present in formula (2.28) and (2.29) acts as a smoothing function. Due to this kind of formulation, the MRF tends to produce a smooth solution: small regions with little support from the neighborhood will be included in bigger clusters.

Finding the exact solution of equation (2.26) can be computationally intractable, but good results are obtained by using an approximate solution based on a fast, iterative algorithm called loopy belief propagation (BP) (Pearl [1988]). The standard belief-propagation algorithm updates messages $\mu_{ij}$ from node $i$ to node $j$. The messages are determined self-consistently by the message update rules:

$$\mu_{ij} = \sum_{l_i} \Phi(l_i, y) \Psi_{ij}(l_i, l_j) \prod_{k \in N(i) \setminus j} \mu_{ik}(l_i) \quad (2.31)$$
where $\mathcal{N}(i)$ denotes the nodes neighboring $i$. Furthermore the belief at node $i$ is proportional to the product of the local compatibility function at that node and all the messages coming into the node $i$:

$$\beta_i(l_i) = \frac{1}{Z} \Phi(l_i, y_j) \prod_{j \in \mathcal{N}(i)} \mu_{ij}(l_i)$$

(2.32)

where $Z$ is a normalization constant (the beliefs must sum 1). If we consider the observation $y_j$ fixed and we focus on the joint probability distribution of $l_i$ in equation (2.26), the belief propagation in fact gives the exact marginal probabilities for all the nodes in any single connected graph.

The belief propagation algorithm does not make reference to the topology of the graph that is applied on. Thus it is possible to run it on a graph with loops as the one defined by a Delaunay triangulation. The messages are initialized with an unbiased value and simply iterated until they converge. The approximate beliefs are read from the belief equations when the messages steady state is reached. Even though the convergence it is not guaranteed, the algorithm is often used in literature to solve marginalization problems in graphs with loops (e.g. Freeman et al. [2002]).

2.3.6 Motion Clusters labeling

The convergency of the Markov Random Field defines clusters boundaries, thus a segmentation procedure is needed to separate independent motion blobs in the image.

Only cells classified by the MRF as boundaries and cells laying on the image borders are considered. A graph is defined by that arcs of the Delaunay triangulation that connect couples of these elements. Starting from a node of choice in the graph, marked as a boundary, arcs are followed once, and nodes are marked with their number of visits. As soon as the number of visit for a boundary node reaches 2 a cycle is found. This is done iteratively in the graph. Cycles define clusters with consistent motion.

2.3.7 Motion prediction

The idea is to propagate the obtained motion clusters as hypotheses for the next frame segmentation. Henceforth, propagated hypotheses are used as a prior for the next frame motion clustering. Our approach is to model the motion of each cluster with an affine motion model and to generate a set of motion hypotheses that are compared with the following frame optical flow. If one of these hypotheses matches, it means that with a good approximation the same kind of motion continues in the following frame.

A motion hypothesis is represented by a motion shape descriptor described by the convex hull of a previously detected motion cluster boundary, propagated in the average direction of its optical flow vectors. More in detail, we describe a motion shape descriptor as:

$$s = \{r_{cog}, c_{cog}, \rho_1, \ldots, \rho_d\}$$

(2.33)

where $s$ is composed by the centroid of the convex hull $(r_{cog}, c_{cog})$ and the radii $\rho_i$ connecting it to the boundary points, as shown in Figure 2.12-right. An affine motion model is
CHAPTER 2. MULTIMODAL ROI GENERATION

Figure 2.13: Motion hypotheses generation example for a segmented motion cluster. Motion shape descriptors (yellow to red) are generated by modifying the radii with gaussian noise from the original descriptor (in light violet). The hypotheses are placed along the line described by the motion model.

computed, with least squares fitting, on the optical flow vectors contained in each motion cluster.

A set of future motion hypothesis shapes $S = (s'_1, \ldots, s'_q, \ldots, s'_d)$ are generated by adding noise to the radii of each motion shape descriptor $s_i$:

$$\rho'_j = \rho_j + \omega_{\rho} \quad \rho_j \in s_i$$  \hspace{1cm} (2.34)

where $\omega_{\rho}$ represents a sample from a gaussian distribution with mean 0 and variance $\sigma_{\rho}$.

The position of a generated hypothesis $s'_i$, relative to a motion cluster, is then located in the image by using the following formula:

$$\begin{bmatrix} r'_{cog} \\ e'_{cog} \end{bmatrix} = \begin{bmatrix} ||\omega_r \hat{v}_{V_i}|| \cos(\delta_{V_i}) \\ ||\omega_r \hat{v}_{V_i}|| \sin(\delta_{V_i}) \end{bmatrix} + \begin{bmatrix} r_{cog} \\ e_{cog} \end{bmatrix}$$

where $||\hat{v}_{V_i}||$ represents the average magnitude of the optical flow vectors set $V_i$ present in the motion cluster $i$ and $\delta_{V_i}$ the average angle. The variable $\omega_r$ represents a sample from a gaussian distribution with mean 0 and variance $\sigma_r$. A visual example of motion hypotheses generation is shown in Figure 2.13.
The process of matching between motion clusters hypotheses produced at time \((t - \tau)\) and optical flow vectors at time \(t\) is obtained by firstly selecting the optical flow vectors \(V^*_i\) covered by each hypothesis \(s'_i\) in the new image frame \(t\) with a fast in-polygon test (MacMartin and Stuart [1992]). Then, the motion model associated with \(s'_i\) is applied to the selected vectors \(V^*_i\) of frame \(t\) and a fitting error is computed. The motion model fitting error is calculated as the distance between the end-points of the optical flow vectors \(V^*_i\) and the end-points of the vectors computed by using the motion model associated to \(s'_i\). The motion hypothesis with the smallest distance, less than a minimum threshold, is assigned to the covered cells in the frame at time \(t\). A negative fixed bias \(h_p\) is added in equation (2.25) to the matched cells. Thus, for those cells, the evidence of being part of a coherent motion cluster (label 1) is strengthened. In order to avoid negative values in \(h_l\), each entropy value is under-saturated to the value 0.

### 2.3.8 Experimental Results

The algorithm has been tested on several image sequences taken from PETS2000-PETS2001 standard dataset (Young and Ferryman [2005]) and a real-world outdoor dataset. Image sequences which show close relative motion have been selected from PETS2000 and PETS2001 datasets in order to show significant results.

KLT algorithm has been set to track 800 features each frame. In order to obtain a quantitative result, a sequence motion segmentation detection rate is evaluated. A correct detection is defined as a segmented moving region bounding box that overlaps at least 70% of the area of the annotation box. Manual annotations of moving objects are given from the PETS2000-2001 datasets. Table 2.1 quantifies the performance of the algorithm. The average performance gain obtained by using the presented motion prediction technique (see Section 2.3.7) is in average 4.7% in the four sequences. We remark that the aim of this work is not to track objects but to segment independent image motions. Some images taken from the four evaluation sequences are shown in Figure 2.14 to 2.15.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq1 (people walking)</td>
<td>80.0%</td>
<td>86.96%</td>
</tr>
<tr>
<td>Seq2 (car and pedestrian)</td>
<td>88.0%</td>
<td>95.65%</td>
</tr>
<tr>
<td>Seq3 (car passing)</td>
<td>92.0%</td>
<td>92.0%</td>
</tr>
<tr>
<td>Seq4 (car and pedestrian)</td>
<td>87.5%</td>
<td>84.0%</td>
</tr>
</tbody>
</table>

**Table 2.1:** Performance evaluation on four sequences extracted from PETS2000 and PETS2001 datasets.

The mobile platform, called Smartter (see appendix A for more details), used to acquire the outdoor datasets, is based on a Daimler-Chrysler Smart vehicle equipped with several active and passive sensors. The camera is mounted behind the windscreen and it is equipped with a wide field of view lens. A moving dataset and a static dataset have been retrieved to show the performance of the algorithm in different real-world
CHAPTER 2. MULTIMODAL ROI GENERATION

conditions. The dataset was acquired during a rainy day to demonstrate the functionality of the method in a difficult weather condition. In the dataset 1, the Smarter is moving and several cars are passing in the other lane. The cars are segmented together due to the similar motion; the algorithm prefers to cluster together similar motion than producing overclustering. In a bigger cluster it is more likely to find objects than in many fragmented blobs. In Figure 2.16-left the Voronoi tessellation and local potentials are shown, the overlayed motion clusters are depicted in Figure 2.16-right. In the dataset 2, the mobile platform is waiting at a red traffic light. The car in the other lane is correctly segmented. In Figure 2.17-left the optical flow, the triangulation and local potentials are shown, the overlayed motion clusters are depicted in Figure 2.17-right. The feature tracker tracks 500 features per frame. The Markov random field steady state is always reached in less than 30 message propagations. Computational time is around 0.9s per frame in an non-optimized implementation on a 1.6Ghz Centrino computer.

Figure 2.14: Left: Two segmented pedestrians are moving in the street in different directions. Due to the lack of features the moving convex hull does not cover completely the moving pedestrians. Right: A car and a pedestrian are moving in opposite directions; well defined separation is shown.

Another experiment was performed to test the scalability of the algorithm with respect to the number of extracted features. The optical flow information is consequently decreased but, as shown in Figure 2.18-left, the motion segmentation scales gracefully with respect to the features quantity. The motion segmentation algorithm here described has been also used as a constraint for an AdaBoost based car detection. An Haar feature based Adaboost classifier is trained with a car data set (trunk/front) to obtain car detection. Classically, the trained classifier searches all over the image for classified features at different scales (Viola and Jones [2002]). This extensive search is now constrained in the segmented motion clusters. This enhances the execution speed and reduces the false positive rate because the car is only searched in a relative motion area. The search space is now constrained in an area more than 52% smaller than the original image. The constrained Adaboost detection is shown in Figure 2.18-right.
2.3. MOTION SEGMENTATION USING A LOCAL ENTROPY FIELD

Figure 2.15: Left: A car is moving on a street. A small incorrectly segmented moving region is found (green hull) due to optical flow tracking errors. Right: A car and a pedestrian are moving in opposite directions; well defined separation is shown.

Figure 2.16: Outdoor set 1: moving observer/moving objects. Local entropies field and Voronoi tessellation are shown on the left figure, cells are color coded with respect to the magnitude of the local entropies (from blue to red). Two low entropy zones are colored in blue. The gray area is associated to optical flow with magnitude less than 1 pixel. On the right figure the segmented moving clusters are overlayed on the original image. Two cars in the left lane are segmented together; the right cluster segments the features of the right part of the road. The dotted central cluster depicts a static motion cluster.
CHAPTER 2. MULTIMODAL ROI GENERATION

Figure 2.17: [Outdoor set 2: static observer/moving objects. Local entropies field and Voronoi tessellation are shown on the left figure, cells are color coded with respect to the magnitude of the local entropies (from blue to red). Only the cells associated with left car show a coherent motion. On the right figure the segmented moving clusters are overlayed on the original image. The car in the left lane is correctly segmented.]

Figure 2.18: Left: Scalability of the proposed technique with respect to the number of tracked features. Right: Haar based AdaBoost detection of cars (front/trunk). The classification algorithm is run only in the segmented moving regions. The overlayed yellow boxes describe a car detection. The white car is not detected due to the poor descriptive properties of the Haar feature.
2.4 Conclusions

In this chapter we have shown two applications related to the topic of scene analysis. Both of them focus in the creation of regions of interest. The first, through the processing of 3D range data and omnidirectional images, aims to define regions which might contain artefacts in an outdoor environment. The second instead, processes a sequence of images to segment similar motions. There, a higher level processing, like an object detection method, can be constrained in the interesting area.

Both algorithms are based on probabilistic methods that handle the uncertainty and incompleteness of the information in a real world environment. In the first method we introduced a sensor fusion technique based on Self Organizing Networks and Bayesian modeling. Simple color based clustering, from image, and smoothness voting, from 3D point cloud, are used as main source of information of the presence of artefacts. In the second algorithm, we have employed a Markov Random Field on a measure of local motion uniformity, namely a local entropy field. Moreover we produced multiple motion hypothesis for the following frame in order to enhance motion segmentation of the successive frame. Qualitative and quantitative experiments on standard and real life datasets show very good results.
Multimodal Object Detection in Urban Environments

One of the most debated topics nowadays, in the field of mobile robotics, is how to render a robot’s task safe with respect to human interaction. A lot of efforts have been put lately in the field of vehicle navigation in urban environments (see for example the DARPA 2007 challenge (dar [2007])). The idea of having an intelligent car that can autonomously, and safely, drive the passengers to a selected endpoint destination is appealing for research and industry. To this respect, perception and analysis of the scene is crucial.

The DARPA Urban Challenge was held on November 3, 2007, at the former George Air Force Base in Victorville, California. Teams have been required to build an autonomous vehicle capable of driving in traffic, performing complex maneuvers such as merging, passing, parking and negotiating intersections. This event was truly groundbreaking as the first time autonomous vehicles have interacted with both manned and unmanned vehicle traffic in an urban environment.

There, it was shown that a reliable navigation in urban environments is possible and reliable. Moving objects though were restricted to be just moving cars that have been detected and tracked using often simple, but effective, assumptions (see for example the work of Petrovskaya and Thrun [2008] on model fitting for cars). There is no real analysis of the scene: stops, traffic signs and intersections are encoded in a GPS referenced map. Even though this challenge proved notable advances in navigation research it did not really tackle the problem of scene analysis.

From the different participants in typical urban traffic scenarios, pedestrians are the most vulnerable ones as they are not protected by any kind of safety equipment as they exist for motorists and cyclists. This fact is lamentably reflected in the annual traffic accident statistics, as they are published by several national and international organizations, e.g. the Touring Club Switzerland (tcs [2007]), National Highway Traffic
Safety Administration ([NHTSA]) or Fédération Internationale de l’Automobile (FIA [2008]). The statistics are not encouraging: TCS reports an increase of 14% of pedestrian related accidents in the last three years; NHTSA informs about 4784 pedestrian fatalities in United States during the year 2006, which accounted for 11.6% of the total 42642 traffic related fatalities; FIA reports that pedestrian accidents occur even more frequently in Europe, see Figure 3.1 for a graphical survey.

![Figure 3.1: The Fédération Internationale de l’Automobile (FIA) report on pedestrian accidents (fatalities per million inhabitants).](image)

Of equal dramatic importance is the safety among cars to avoid collisions in different traffic situations. A report of World Health Organization ([who [2009]]) shows that this problem is severe in every country of the world, see Figure 3.2.

Interestingly, a study completed by Rumar [1985] identifies the causes of traffic related accidents, see Figure 3.3. It clearly shows that the car driver play the biggest role in causing road accidents (approximately 92% of the times). Motivated by this information, it is important to create reliable techniques that would help the driver in identifying salient objects in everyday risky traffic situations.

In this chapter we describe our approach in order to detect and track pedestrians and cars in urban environments. Our novel method is based on a multimodal sensor fusion that uses 2D laser rangefinder and camera images as main source of information. By using these two sensors we obtain two different kinds of information: the appearance of the object and a partial geometrical description. The content explained in this chapter has been mainly published in 4 peer-reviewed conference papers: Spinello and Siegwart [2008a], Spinello et al. [2008a,b, 2009b].

### 3.1 A Challenging Topic

Our multimodal method is based on vision and laser range data fusion. In this section we want to briefly highlight the difficult tasks that each of the components has to address...
3.1. A CHALLENGING TOPIC

**Figure 3.2:** Annual number of road fatalities each 100000 inhabitants (who [2009]).

in order to obtain good performances.

Pedestrians are particularly difficult to detect because of their high variability in appearance due to clothing, illumination and due to the fact that the shape characteristics depend on the view point. In addition, occlusions caused by carried items such as backpacks, as well as clutter in crowded scenes can render this task even more complex by dramatically changing the shape of the object. Given the overall vertical-like object structure, detection systems can be easily fooled in urban environments by poles, signs and by buildings architectural features like windows and doors. Pedestrians are described, when using a laser rangefinder, by just a few points immersed in a complex scene; moreover the shape characteristics change with respect to self occlusions (e.g. leg with leg occlusion) or change with respect to the distance to the observer due to limited sensor angular resolution (point density change). Limited laser angular resolution restricts the detection of pedestrians that are very far to employ vision-only techniques. These techniques are also limited by the fact that the informative content present in a small area, described by just few pixel, is also very low.

Even though the visual appearance of vehicles is less variable than the complex appearance of pedestrians, cars remain a challenging object category to detect. View points also play an important role: lateral views are totally different from frontal or rear views. If scanned by a laser, vehicles often appear as lines or corner segments that are hard to distinguish from the background.

Tracking such kinds of objects also represents a challenging topic. It is very hard, if not virtually impossible, to determine a mathematical formulation of a pedestrian motion model, due to the complete freedom of movement. Car motion model instead, can be formalized through the work of Ackermann [1818] that defines a precise kinematic model.
CHAPTER 3. MULTIMODAL OBJECT DETECTION

Figure 3.3: Principal causes of road accidents (Rumar [1985]). It is interesting to notice that in approx. 92% of the times, the car driver is being part of the accident causes.

of a steering vehicle. Creating a system able to produce an accurate tracking of such different motion patterns represents a hard challenge.

3.2 State of The Art

In this section we refer the works related to laser and camera based detection of pedestrians and cars as well as multisensor detection methods and object tracking.

Several approaches can be found in the literature to identify a person in 2D laser data. A popular approach is to extract legs by the detecting moving blobs that appear as local minima in the range data (Fod et al. [2002], Scheutz et al. [2004], Schulz et al. [2003]). Geometrical and motion features have been used. If just motion features are used people that do not move cannot be detected. The work of Topp and Christensen [2005] overcomes this problem and it is able to obtain good results in an indoor, cluttered, environment. Hähnel et al. [2003b] considered the problem of classifying beams in range scans that are reflected by dynamic objects. An expectation maximization (EM) estimation is run in order to determine which beam has been reflected by a dynamic object as a person. The work of Xavier et al. [2005] is based on geometrical features, they segment the range scan into clusters and apply a set of heuristics in order to distinguish between lines, circles and legs. A significantly important work has been produced by Arras et al. [2007] that employs a learning approach to cluster the laser data and learn an AdaBoost classifier from a set of geometrical features extracted from the clusters. Another approach that obtained very good results has been presented by Luber et al. [2008], that was able to detect and track several classes of objects through the unsupervised creation of exemplar models. Attempts to detect pedestrians in 3D range data have been made by L. Navarro-Serment and Hebert [2009] who use PCA analysis and geometrical descriptors classified by Support Vector Machines. In the paper of Petrovskaya and Thrun [2008] tracking and detection of multiple vehicles using laser is obtained through a model based approach. It encompasses both geometric and dynamic properties of the tracked vehicle in a single Bayes filter. Other range data related approaches based on segmentation and classification are Zhao and Thorpe [1998] and Streller et al. [2002]. The first enforces a rectangular model
of a car in range data by using heuristics on extracted lines and it performs an Extended Interactive Motion Model filter tracking. In the latter several motion models are built in order to be applied to simple geometrical models of vehicles.

In the area of image-based people detection, there mainly exist two kinds of approaches (see Gavrila [1999] for a survey). One uses the analysis of a detection window or templates (Gavrila and Philomin [1999], Viola et al. [2003]), the other performs a parts-based detection (Felzenszwalb and Huttenlocher [2000], Ioffe and Forsyth [2001]). The detection window approach uses a scalable window that is scrolled through the image. For each step a classification of the image area under the detection window is obtained. A template based detection technique is similar to the previously described approach, but in this case a simple distance measure is performed between the edges present in the image under the template silhouette and the silhouette itself. A part based detection method aims at independently detecting parts in order to form hypothesis of entire objects. There exist plenty of computer vision based pedestrian detection systems available in literature. We here refer to the most successful. Leibe et al. [2005] presented an image-based people detector using Implicit Shape Models (ISM) with excellent detection results in crowded scenes. This method is based on a database of bag of words, extracted from standard descriptors, that vote for objects centers. A mean shift mode estimation is used to define object hypotheses in the continuous space and a maximum descriptor length to select the winning ones. This method deeply inspired the work on the vision detector explained in Section 3.8.3. Noticeably also Dalal and Triggs [2005] presented an image based human detection algorithm with very good detection results and remarkable performance in terms of execution speed. This method is based on the classification of special image descriptors, Histogram of Oriented Gradients (HOG), computed over blocks of different sizes and scales in a fixed size detection window. This descriptor is based on a collection of normalized image gradients on each cell. The resulting high dimensional vector is then classified with a linear support vector machine (SVM). Zhu et al. [2006] then refined this detector by using a fast rejector-based SVM cascade to discriminate the presence of a human in the detection window. Car and pedestrian detection using computer vision has been shown by Leibe et al. [2007] with a multiclass ISM based detector. Ess et al. [2009] presented a stereo camera detection and tracking system based on an HOG classifier for cars and humans in cluttered urban environments.

Existing people detection methods based on camera and laser rangefinder data either use hard constrained approaches or hand tuned thresholding. Cui et al. [2005] use multiple laser scanners at foot height and a monocular camera to obtain people tracking by extracting feet and step candidates. Zivkovic and Kröse [2007] use a learned leg detector and boosted Haar features extracted from the camera images to merge this information into a parts-based method. However, both the proposed approach to cluster the laser data using Canny edge detection and the extraction of Haar features to detect body parts is hardly suited for outdoor scenarios due to the highly cluttered data and the larger variation of illumination encountered there. Schulz [2006] uses probabilistic exemplar models learned from training data of both sensors and applies a Rao-Blackwellized particle filter (RBPF) in order to track the person’s appearance in the data. The RBPF tracks
CHAPTER 3. MULTIMODAL OBJECT DETECTION

countours in the image based on Chamfer matching as well as point clusters in the laser scan and computes the likelihood of different prototypical shapes in the data. However, in outdoor scenarios lighting conditions change frequently and occlusions are very likely, which is why contour matching is not appropriate. Moreover, the RBPF is computationally demanding, especially in crowded environments. B. Douillard [2008] employs a Conditional Random Fields (CRF) learned on 2D laser data and image features to detect multiple classes of objects (cars, pedestrian, vegetation). Very good results are obtained but occlusions and overlapping hypothesis are not handled. Wender and Dietmayer [2008] use camera and 3D laser to detect and track multiple cars in front of the vehicle. Premebida et al. [2009] did not use tracking but evaluated several standard vision and laser detectors with several centralized and decentralized fusion rules.

Several methods have been proposed to track moving objects in sequential data (see Cox [1993] for an overview). The most common ones include the joint likelihood filter (JLF), the joint probabilistic data association filter (JPDAF), and the multiple hypothesis filter (MHF). Unfortunately, the exponential complexity of these methods makes them inappropriate for real-time applications such as navigation and path planning. Cox and Miller [1995] approximate the MHF and JPDA methods by applying Murty’s algorithm and demonstrate in simulations the resulting speedup for the MHF method. Rasmussen and Hager [2001] extend the JLF, JPDA, and MHF algorithms to track objects represented by complex feature combinations. Schumitsch et al. [2006] propose a method to reduce the complexity of MHT methods introducing the Identity Management Kalman Filter (IMKF) for entities with signature. Arras et al. [2008] uses a multi hypothesis tracking to adaptively address the problem occlusions and self occlusions when tracking multiple pedestrians. An interesting related work in the field of pedestrian tracking is the method developed by Lau et al. [2009] to track groups of people by employing clustering by distance incorporating it in a multiple hypothesis tracking system.

In the specific field of using Kalman Filters and Bayes theory for obtaining sensor fusion it is important to cite the works of Crowley [1985, 1989] regarding robot navigation and world modeling, the work of Durrant-Whyte [1987] on consistently integrating and rejecting spurious measurements, the seminal work on Simultaneous Localization and Mapping (SLAM) of Leonard and Durrant-Whyte [1991] and the work of Crowley and Demazeau [1993] on techniques for sensor data fusion.

With the proposed methods we overcome the limitations associated to single sensor setups. Camera based methods, even though very robust, have the drawback of requiring a sufficient image contrast to work efficiently and of being dependent on the camera-lens for the size of the field of view. Laser rangefinder based methods have the drawback of obtaining just few points per scan, therefore objects are described by small sets of points. They function without environment light and in virtually every weather condition. Motivated by these reasons, we tailored a multimodal approach that probabilistically combines an image based detector with a range based detector. Specifically, we addressed the limitations of ISM by producing extensions that create better explanation of the feature distribution and of the subparts of objects. In range data we addressed the problem of feature locality by taking into account data neighborhoods. We reduced the computational
requirements of multihypothesis tracking systems (MHT or JPDA) by introducing a less
demanding tracking method based on multiple motion models able to scale well with the
number of objects to track.

3.3 Structure of the Chapter

This chapter is divided in 5 sections. In the next section we explain the overall scheme
of our detection and tracking system, in Section 3.6 the laser based detection systems. In
section 3.8 we explain the camera based detection systems. In Section 3.9 we show the
fusion and tracking methods. In the last section of the chapter we show experimental
results.

3.4 Overview of the developed techniques

Our pedestrian and car detection system is based on a multimodal processing of monocular
camera images and 2D laser data. The system is trained on a set of manually labeled
camera and laser data. It is possible to define three separate blocks of processing for this
method: laser-based detection to obtain structure information, camera-based detection to
obtain appearance information and finally sensor fusion.

The laser-based detector groups points in clusters and classifies them. Two methods
have been developed: in the first developed techniques (Spinello and Siegwart [2008a],
Spinello et al. [2008a,b]) line segments are extracted from the range data. Then a Delaunay
triangulation connects the centers of the cluster to perform an additional segmentation
in order to reduce overclustering. The clusters are classified with an AdaBoost cascade
(Freund and Schapire [1997]) of linear SVMs in order to distinguish between pedestrian
and background. In Spinello et al. [2009b] a step further is accomplished in order to detect
multiple classes, namely cars and pedestrians. There a Conditional Random Field (CRF)
(Lafferty et al. [2001]) is run on the Delaunay graph. The node features are obtained from
a multiclass stump-based AdaBoost preclassification; edge features tend to enforce that
nodes at small distance and same class are classified with the same label. This 2D laser
classification technique produces a sound probabilistic theory generalizable for several
categories of objects.

Two main techniques have been developed for the camera-based detection system.
The first method, employed in Spinello and Siegwart [2008a], is based on the classification
of Histogram of Oriented Gradients descriptors computed in a scalable detection window
(Dalal and Triggs [2005]). In order to obtain a fast classification an AdaBoost cascade of
SVMs has been used. Another approach, based on Implicit Shape Models (ISM), has
been taken in Spinello et al. [2008a,b, 2009b] in order to reduce the drawback of training
with a big amount of images and to control, with more flexibility, occlusions and shape
variances. A codebook, a database composed of standard image descriptors learned
from a training set, is used to match features and to vote for object centers. Several
improvements, called ISMe, have been produced in order to ease the feature selection
and weighting, hypothesis generation and managing explicit subparts. This method has
been refined in Spinello et al. [2009b], in order to detect multiple categories of objects. In opposition to a computer vision-only methods we make use of the depth information coming from the laser in order to restrict the search space of objects in image scale and space: the algorithms are run just in areas where laser points are present.

In our approach we fused information and tracked objects in laser space. We address the problem of tracking by using the Kalman Filter (KF) theory. For each track multiple motion models are managed in order to handle complex movements of detected objects like pedestrians and cars. The KF-based tracker is also a mean of integrating the evidence coming from the laser detector and the image detector. The two sensor modalities are then combined using an KF sensor fusion modeling. In an earlier work, developed in Spinello and Siegwart [2008a], no tracking system has been included and a Bayesian sensor fusion model has been employed to combine detection likelihoods.

### 3.5 Contributions

With this work we bring together the field of mobile robotics based on laser data processing and the field of computer vision. To the author’s best knowledge this work represents the only multimodal and flexible trainable system, not based on hand tuning or given heuristics, that has been developed. The advantages of using camera and laser sensor fusion principally are:

- direct, precise and instantaneous distance measurement of the detection (due to laser).
- sensors with complementary characteristics (see table 3.1): a moving complex object, deformable in case of a pedestrian, can be described with a high confidence by one sensor or the other.

The contributions introduced are as follows:

- Increasing the featureset of Arras et al. [2007] for laser-based pedestrian detection, considering also n-dimensional features (Spinello and Siegwart [2008a]).
- The implementation of a novel fast graph-cut based segmentation for range data in order to reduce overclustering. (Spinello and Siegwart [2008a])
- A common classification framework for laser and camera data based on a cascade of linear SVMs. (Spinello and Siegwart [2008a]).
• **ISM:** several improvements to the image based object detector ISM, based on the work of Leibe et al. [2005]. The improvement consists in features weighting (Spinello et al. [2008b]), selection (Spinello et al. [2009b]) and matching (Spinello et al. [2008a]). We introduced hypotheses cost function for selection (Spinello et al. [2008a]), an image reasoning method to remove false hypothesis (Spinello et al. [2008b]), explicit subparts reasoning (Spinello et al. [2009b]), object category templates for reducing outliers (Spinello et al. [2009b]) and multiclass handling capabilities (Spinello et al. [2009b]).

• A tracking algorithm based on KF with multiple motion models. The filter can be asynchronously updated with the detection results from the laser and the camera. (Spinello et al. [2008a])

• The use of a 3D scanning device, which facilitates a fast and robust detection of the ground plane and thus helps to disambiguate possible detections of pedestrians. (Spinello et al. [2008b])

• The novel application of boosted Conditional Random Fields (CRF) on laser scans data for classifying multiple classes of objects (Spinello et al. [2009b]). Conditional Random Fields use the AdaBoost preclassification stage for defining local potentials and edge features for defining neighborhood potentials in order to detect cars and pedestrians in complex range data.

### 3.6 Structure Information from Laser Data Analysis

We here describe the steps needed in order to detect objects, namely pedestrians and cars, in the laser scan data (see Figure 3.5). We developed two kinds of detectors and both are based on a supervised learning approach. We assume that the robot is equipped with a laser range sensor that provides 2D scan points \(x_1, \ldots, x_N\) in the laser plane. We start by clustering data points in segments, then we classify segments by applying an AdaBoost classifier (Spinello and Siegwart [2008a]) or by using a Conditional Random Field approach (Spinello et al. [2009b]). The logical steps taken for obtaining object classification are shown in Figure 3.4.

#### 3.6.1 Preparing Range Data: Clustering

A preprocessing step before range data classification has been defined in order to prepare data for the classifiers. In our case we segment a line scan in contiguous groups of points that define segments. Then, we connect them together by using a Delaunay triangulation. This process defines a graph that is used as a mean to obtain grouping between connected segments (Spinello and Siegwart [2008a], Spinello et al. [2008a,b]), in order to reduce overclustering, or as the network for Conditional Random Fields (Spinello et al. [2009b]).
CHAPTER 3. MULTIMODAL OBJECT DETECTION

Figure 3.4: Overview scheme for object classification of range data. Two main approaches have been employed: AdaBoost classification of clustered segments (contoured in blue) and Conditional Random Fields on range data (contoured in red).

3.6.1.1 Creating Segments

Jump Distance Clustering (JDC) is a widely used method for 2D laser range data in mobile robotics (see Premebida and Nunes [2005] for an overview). It is fast and simple to implement: if the Euclidean distance between two adjacent data points exceeds a given threshold, a new cluster is generated, see Figure 3.6. Although this approach performs well in indoor scenarios, it tends to produce poor results for outdoor data. The outdoor environment is geometrically more complex and wider than an indoor scenario. Moreover, dramatic reflections and direct sunlight effects contribute to measurement errors and missing points in the linescan. This often leads to an over-segmentation of the linescan data, fractioned in many small clusters. Clustering is performed with threshold \( \theta_S \). Each cluster \( \tilde{S}_i \) is defined by its left border \( x_{l_i} \), its centroid \( \hat{x}_i \), and its right border \( x_{r_i} \):

\[
\tilde{S}_i = \{ x_{l_i}, \hat{x}_i, x_{r_i} \}
\]  

(3.1)

We define also \( S_i \) as the complete set of points, ordered by the scanning angle, associated to the cluster \( \tilde{S}_i \).

3.6.1.2 Reducing overclustering

To address the problem of oversegmentation we aim to smartly define adjacency among laser segments. Several methods exist in literature to define neighborhoods in metric data (e.g. square cells space subdivision (Tay et al. [2008], Thrun et al. [2005]), Voronoi graphs (Friedman et al. [2007])). We use a Delaunay triangulation, the dual of the Voronoi tessellation, that connects the centers \( \hat{x}_i \) of the segments \( S_i \).

We developed, in Spinello and Siegwart [2008a], a simple yet effective technique that extends the classic Jump Distance Clustering segmentation. It consists in the following steps:
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

Figure 3.5: An urban environment with cars, pedestrian and other objects as it is perceived by a 2D laser. Laser beams are shown in red, circles represents the measured points. Gray beams indicate out of range data due to material reflections, sun related effects and particular object poses.

1. Perform jump distance clustering (see Section 3.6.1.1).

2. Compute a Delaunay triangulation that connects the cluster centers $\hat{x}_i$.

3. Annotate each edge $e_{ij} := (\hat{x}_i, \hat{x}_j)$ of the Delaunay graph with the minimal Euclidean distance between $\tilde{S}_i$ and $\tilde{S}_j$ defined by $d_{ij} = \arg\min(\|\tilde{S}_i - \tilde{S}_j\|)$

4. Remove edges with a distance greater than $\vartheta$ and merge each remaining connected component into a new cluster.

Note that the same threshold $\vartheta_S$ is used twice: first to define the minimum jump distance between the end points of adjacent clusters and then to define the Euclidean distance between clusters. A visual explanation is given in Figure 3.7.

Experimental results have shown that this method, named shortly Delaunay JDC, reduces the cluster quantity of 25% – 60%, significantly reducing overclustering. The additional computational cost due to the Delaunay triangulation and distance computation is lower compared to a full 2D agglomerative clustering approach.
3.6.2 Characterize Range Data: Feature Computation

In order to classify laser data, distinctive features are needed on segmented point clusters. Our approach extends the feature set introduced by Arras et al. [2007]: a collection of geometrical and statistical values computed for each segment and used as input for an AdaBoost classification algorithm. Moreover, we introduce n-dimensional feature descriptors.

We define a feature as a function \( f : S_i \rightarrow \mathbb{R}^n \) that takes the point set contained in a cluster \( S_i \) as an input argument and it returns a real value.

We consider measures of geometrical properties:

- **Width**: this feature measures the Euclidean distance between the leftmost and the rightmost point of the cluster \( S_i \): \( f_1 = ||x_1 - x_n|| \)

- **Number of points**: \( f_2 = ||S_i|| \)

- **Circularity**: this feature measures the circularity of the cluster points and corresponds to the residual sum of squares of a circle fitted into the cluster \( S_i \) in the least squares sense. Circle parameters center \((x_c, y_c)\) and radius \( r_c \) are computed and feature is obtained as: \( f_3 = \sum_{j \in S_i} \left( r_c - \sqrt{(x_c - x_j)^2 + (y_c - y_j)^2} \right)^2 \)

- **Linearity**: this feature corresponds to the residual sum of squares of a line fitted into the cluster \( S_i \) in the least squares sense. The points are converted in polar coordinates

**Figure 3.6**: *Left*: Visual explanation of Jump Distance Clustering (JDC): two consecutive points distant less than the threshold \( \theta_S \) are grouped together. The distance measuring order between points is shown in green. The orange line indicates points distant less than the threshold. *Right*: Resulting JDC clustering of the scene. Orange lines depict points grouped together.
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

and line parameters \((\alpha, r)\) are computed by using the closed form equation suggested in Arras et al. [2007].

\[
f_4 = \sum_j \|S\| (x_j \cos(\alpha) + y_j \sin(\alpha) - r)^2
\]

- **Boundary length:** this feature measures the length of the polyline traced between points of the cluster \(S_i\). We compute the set of subsequent point distances \(D = (d_1, \ldots, d_{n-1})\) where \(d_i = \|x_j - x_{j+1}\|\). Therefore \(f_5 = \sum_j d_j\)

- **Boundary regularity:** this feature measures the standard deviation of the distances of the adjacent points of cluster \(S_i\). If \(\hat{d}\) is the mean of \(D\) then the feature is computed as:

\[
f_6 = \sqrt{\frac{1}{n-1} \sum_j \|\mathcal{B}\| (d_j - \hat{d})^2}
\]

- **Mean angular difference:** this feature is obtained by averaging the angles between the vectors constructed by connecting two consecutive points of cluster \(S_i\). The set \(\mathcal{B} = (b_1, \ldots, b_{n-1})\) is obtained by collecting \(b_i = \angle(x_i - x_{i+1})\) values. \(f_7 = \sum_j \|\mathcal{B}\| b_i \|\mathcal{B}\|\)

- **Mean curvature:** this feature measures the approximated curvature of cluster \(S_i\) by using the suggested formula in Arras et al. [2007]. The set \(\mathcal{C} = (c_1, \ldots, c_n)\) collects the discrete curvature boundaries, computed as \(c_i = \frac{1}{\|x_{i-1}-x_{i+1}\|\|x_{i+1}-x_{i+2}\|\|x_{i}-x_{i+2}\|}\). Feature value is computed as: \(f_8 = \sum_j \|\mathcal{C}\| c_j \|\mathcal{C}\|\)

- **Quadratic spline fitting:** this feature measures the residual sum of squares of a quadratic B-Spline regression \(spl_2\) (a piecewise polynomial approximation introduced by Boor [1978]) of the points in the cluster \(S_i\). \(f_9 = (spl_2(x_i, y_i) - y_i)^2\)

- **Cubic spline fitting:** this feature measures the residual sum of squares of a cubic B-Spline regression \(spl_3\) of the points in the cluster \(S_i\). \(f_{10} = (spl_3(x_i, y_i) - y_i)^2\)

We consider also measures of statistical dispersion (cluster compactness):

- **Standard deviation with respect to centroid:** in a cluster \(S_i\) the feature is computed as:

\[
f_{11} = \sqrt{\frac{1}{n-1} \sum_j \|S\| (x_j - \hat{x})^2}
\]

- **Average deviation from median:** the median of a finite list of numbers can be found by arranging all the observations from lowest value to highest value and selecting the middle one. If there is an even number of observations, the median is not unique, so the mean of the two middle values is taken. Using the approach suggested in Arras et al. [2007], we sort each dimension of the 2D data in the cluster \(S_i\) independently, by considering each dimension uncorrelated to the other, and we compute the median in 2D as \(\hat{x} = (\hat{x}, \hat{y})\). The feature is therefore computed as \(f_{12} = \frac{1}{\|\mathcal{S}\|} \sum_j \|x_j - \hat{x}\|\).

- **Kurtosis with respect to centroid:** kurtosis defines the degree of peakedness of a distribution. It is defined as the fourth standardized moment of the cluster \(S_i\).

\[
f_{13} = \frac{\sum_j \|x_j - \hat{x}\|^4}{\|\mathcal{S}\| f_1^4}
\]
Figure 3.7: **Left:** A Delaunay triangulation is built on the centers of the segments. This defines topology among segments. **Right:** An agglomerative clustering is run on the triangulation in order to group segments connected with arcs $d_{ij} < 8$ (in green).

- **Radius:** this feature is represented by the radius $r_c$ of the fitted circle on the segment $S_i$ computed in feature $f_5$. Feature is computed as $f_{14} = r_c$.

- **PCA ratio:** this feature describes the ratio between the second biggest eigenvalue $\lambda_2$ and the biggest eigenvalue $\lambda_1$ of the cluster $S_i$. It describes a measure of the aspect ratio of the principal axis aligned bounding box. This feature is computed as: $f_{15} = \frac{\lambda_2}{\lambda_1 + 1}$.

- **Bounding Box area:** this feature $f_{16}$ represents the area of the bounding box on the cluster $S_i$.

In addition we consider cluster descriptors:

- **$Q$-binned histogram descriptor:** the bounding box containing the points that belong to the segment $S_i$ is considered for a binary tessellation composed of $Q = R \times L$ rectangles. Each cell is flagged with the number of points that belong in that cell. The feature is described by a set $F_A = (f^a_1, \ldots, f^a_q)$, where $f^a_i$ represents the quantity associated to the $i$-bin.

- **PCA $Q$-binned histogram descriptor:** before computing $F_A$ a PCA analysis is run in order to find the axis in which the statistical dispersion is bigger. Such axis is used to align the bounding box. Feature binning is the same of $F_A$, therefore $F_B = (f^b_1, \ldots, f^b_q)$, where $f^b_i$ represents the quantity associated to the $i$-bin.

- **Mean curvature binned descriptor:** the curvature distribution set $C$ is accumulated in a histogram of $Q^C$ bins to define the feature $F_C$. 
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

- **Boundary regularity binned descriptor:** the boundary length distribution described by the set $D$ is accumulated in a histogram composed of $Q^D$ bins in order to define the feature $F_D$.

We then consider the convex hull of the cluster $S_i^H$ and compute the before mentioned features by using only the hull points: Convex Hull Circularity, Convex Hull radius, Convex Hull Linearity, Convex Hull Boundary length, Convex Hull Boundary regularity, Convex Hull Mean angular difference, Convex Hull Mean curvature, Convex Hull quadratic and cubic spline fitting, Convex Hull area.

We then compute the distance of the cluster $S_i$ with respect to the laser sensor origin $d = \| \hat{x} \|$. We use the quantity $d$ to produce a group of 2D features by combining it to some geometrical and statistical features: $F_1 = (f_1, f_d), F_2 = (f_2, f_d), F_3 = (f_3, f_d), F_4 = (f_4, f_d), F_5 = (f_5, f_d), F_6 = (f_6, f_d), F_7 = (f_7, f_d), F_8 = (f_8, f_d), F_9 = (f_9, f_d), F_{10} = (f_{10}, f_d), F_{11} = (f_{11}, f_d), F_{12} = (f_{12}, f_d), F_{13} = (f_{13}, f_d), F_{14} = (f_{14}, f_d), F_{15} = (f_{15}, f_d)$.

Therefore for each segment $S_i$ a set of features $F' = (f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, F_A, F_B, F_C, F_D), a vector of convex-hull associated features $F''$ and a set of 2D features $F^* = (f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15})$ are computed.

3.6.3 Classifying Range Data: Feature Classification

Our approach is totally based on supervised learning techniques. Given a manually labeled dataset we run a training phase for a classification method in order to produce, without human intervention, a statistical model of the objects based on data. In order to classify or preclassify laser data based on the feature set $F'$, $F''$ and $F^*$ we used AdaBoost or its computing-time optimized variant, AdaBoost cascade. Our laser detection approach is largely based on these techniques, therefore in this section, we explain AdaBoost and its theoretical advantages.

3.6.3.1 Classifying range data with AdaBoost

We here briefly describe the training and classification stage of AdaBoost. AdaBoost, short for Adaptive Boosting, is a machine learning algorithm, formulated by Freund and Schapire [1997]. It is an ensemble machine learning method: an ensemble of classifiers is a set of classifiers whose individual decisions are combined in a weighted or unweighted voting approach to classify new examples. A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is that the classifiers are **accurate** and **diverse** (see L. K. Hansen [1990]). An **accurate** classifier is one that has an error rate < 50%. Two classifiers are **diverse** if they make different errors on new samples. Intuitively if we combine **diverse** classifiers the errors made by the classifiers are independent and a majority vote may correctly classify the sample.

The advantages of using ensemble classifiers are three folds: statistical, computational and representational:

**Statistical**: a learning algorithm can be interpreted as a search of the best hypothesis in the space $H$ of all the hypotheses. The statistical problem arises when the amount
of training data is small with respect to the size of the hypothesis space. Without sufficient data, the learning algorithm can find many different hypotheses in $\mathcal{H}$ that all give the same accuracy on the training data. By computing an ensemble out of all accurate classifiers the algorithm can weight their vote and reduce the risk of choosing the wrong classifier (see Figure 3.8-left).

**Computational**: many learning algorithms work by performing a local search that may get stuck in local minima (e.g., neural networks, decision trees). When the cardinality of the training data is sufficiently big, the statistical problem does not arise and obtaining the best hypothesis may still be very difficult computationally (optimal training for neural networks and decision trees is NP hard (Blum and Rivest [1992], Hyafil and Rivest [1976])). An ensemble built by running the local search from many different starting points may provide a better approximation to the true unknown function than any of the individual classifiers (see Figure 3.8-middle).

**Representational**: in most applications the true function cannot be represented by any of the hypotheses in $\mathcal{H}$. By forming weighted sums of hypotheses drawn from $\mathcal{H}$ it may be possible to expand the space of representable functions. In many learning algorithms $\mathcal{H}$ is the space of all possible classifiers; nonetheless with a finite training set these algorithms will explore only a finite set of hypotheses and they will stop as soon as one fits the training data (see Figure 3.8-right).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.8.png}
\caption{The three key advantages of using ensemble classifiers, from left to right: statistical, computational and representational. The black dotted line represents the set of the hypotheses $\mathcal{H}$. Hypotheses are plotted as blue dots, the true function $z$ in red.}
\end{figure}

Hence the ensemble methods in general, and AdaBoost in particular have the promise of reducing these three shortcomings of standard learning algorithms. AdaBoost is used in conjunction with other learning algorithms (weak learners) to improve their overall performance. AdaBoost is adaptive in the sense that subsequent classifiers are tweaked
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

**Algorithm 1** Train an AdaBoost classifier

**Require:** $(b_1, l_1) \ldots (b_m, l_m)$ where $b_i \in \mathcal{B}$ and $l_i \in (-1, 1)$

1. A weight represented by a real number is assigned to each sample. The weight vector, for the first step, is initialized as $w^0_1 \ldots w^0_m = \frac{1}{m}$
2. $t_e \leftarrow$ # of classifiers for AdaBoost
3. for $t = 1$ to $t_e$ do
4. Train a classifier $h_t : \mathcal{B} \rightarrow \{-1, +1\}$ that minimizes the error with respect to the distribution $w^t$: $h_t = \arg \min_{h \in \mathcal{H}} (e_t)$, where $e_t = \sum_{i=1}^{m} w^t_i [l_i \neq h(b_i)]$
5. $e_t$ is the error rate of the classifier $h_t$
6. if $e_t < 0.5$ then
7. Choose $\alpha_t = \frac{1}{2} \ln \left( \frac{1-e_t}{e_t} \right)$
8. The weight vector for the samples is updated in order to be a probability distribution: $w^{t+1}_i = \frac{w^t_i e^{-\alpha_t h_t(b_i)}}{\sum_j w^{t+1}_j}$
9. end if
10: end for
11: The AdaBoost classifier output is given by $H(b_i) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(b_i) \right)$

in favor of those instances misclassified by previous classifiers. It is sensitive to noisy data and outliers but it is less susceptible to the overfitting problem than most learning algorithms (see Dietterich [2000]). Formally, AdaBoost selects a set of weak classifiers (better than chance classifiers) and combines them linearly into a final strong classifier, see Algorithm 1.

The weight sample distribution $w_t$ is updated in line nr. 8 of Algorithm 1 and follows the rule that if the sample $i$ is correctly classified then $w^{t+1}_i < 1$ otherwise it is bigger than 1. Thus, after selecting a classifier $h_t$ for the distribution $w_t$, the examples $b_i$ that the classifier $h_t$ identified correctly are weighted less and those that it identified incorrectly are weighted more. Therefore in subsequent iterations AdaBoost constructs progressively more difficult learning problems. Particularly the classifier $h_t$ can be constituted by whichever *weak classifier*: a decision stump (single axis parallel partition of the space), a multi layer perceptron (Bishop [1996]) (general non linear function approximators), a radial basis function (Buhmann [2003]) (non linear function approximations based on kernels), a decision tree (Breiman et al. [1984]) (hierarchical partition of the space), etc. In order to classify a new unseen input vector $b_i$ just the line nr. 11 of Algorithm 1 has to be computed.

### 3.6.3.2 AdaBoost with weighted soft SVMs

Every work related to pedestrian and car detection explained in this thesis employs AdaBoost as a classification mean, or as a preclassification stage, for detecting objects in the laser data. In the work of Spinello and Siegwart [2008a], Spinello et al. [2008a,b] we employed Support Vector Machines (SVMs) (Cortes and Vapnik [1995]) as weak learners. Support vector machines are a set of supervised learning methods used for classification
and regression. SVM considers two classes of input data as a set of vectors in an n-dimensional space. It aims to compute an optimal separating hyperplane between the two data sets. There are many hyperplanes that classify the data. The maximum separation (or margin) between the two classes is usually desired: the selected hyperplane is the one that maximizes the distance to the nearest data point on each side of the hyperplane. Intuitively, the larger the margin the lower the generalization error of the classifier. This guarantees a good separation and generalization. If such a hyperplane exists it is known as the maximum-margin hyperplane. We used SVMs as flexible components of AdaBoost: it is used as an optimal stump classifier by applying it to each single feature in the feature set $F'$ and $F''$, or it is used as a multidimensional classifier for each feature contained in the set $F^*$. More precisely, each data sample is composed of $b_i = F^* = (F', F'', F^*)$, where each element of $F', F''$ is considered as an independent dimension, therefore it is processed with 1D linear SVMs; each feature contained in $F^*$ is instead classified with 2D linear SVMs. We used soft margin SVM (also called C-SVM) in order to find the linear separating hyperplane also in case that no linear separation exist by introducing slack variables, which measure the degree of misclassification of a data point. The constraints of a soft margin SVM, for $m$ multidimensional inputs $b_i$ with labels $l_i$, are defined as:

$$l_i(\mathbf{v} \cdot \mathbf{b}_i - g) \geq 1 - \xi_i \quad 1 \leq i \leq m$$

(3.2)

where $\mathbf{v}$ is the normal vector perpendicular to the hyperplane and the parameter $g$ is the offset of the hyperplane from the origin along the normal vector $\mathbf{v}$. The aim is to use the constraint of formula (3.2) to minimize $(\mathbf{v}, m, \xi)$ by using quadratic programming (QP):

$$\frac{1}{2}||\mathbf{v}||^2 + C \sum_i \xi_i$$

(3.3)

Therefore the optimization is a trade off between margin and error penalty.

We employed SVMs as weak learners for classifying features contained in $F^*$ in order to find relations between pair of variables. Even though geometrical properties (e.g. circularity or linearity) of an object do not change with respect to the distance from the observer, the data retrieved with a laser range sensor is affected by angular discretization effects and reflections that increase their influence with respect to the distance. With the classification of the set $F^*$ we aim to find, if they exist, linear correspondences between observed distance and statistical/geometrical features. It can argued that this problem can be solved just by using standard decision stump classifiers. The advantage of using SVMs is that potentially less weak classifiers would be used for the same AdaBoost classification problem and a better feature space segmentation can be achieved due to the higher degrees of freedom of the SVM hyperplane. A explanatory visual example is shown in Figure 3.9.

In order to use SVMs as components of the AdaBoost algorithm we need to modify the formula (3.3) in order to take in account the data weights of the AdaBoost algorithm. We need to formulate a constrained optimization like the one of (3.3) that takes in account
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

Figure 3.9: **Left**: two class labeled data is shown in blue and orange. **Middle**: Hyperplane subdivision in case of AdaBoost with decision stump components. **Right**: Hyperplane subdivision in case of AdaBoost with linear kernel SVM components. The latter case obtains the same classification with less weak learners.

weights for data instances. The idea is to weight the slack variable with $w_i$:

$$l_i(v \cdot b_i - g) \geq 1 - w_i \xi_i \quad 1 \leq i \leq m$$

$$\frac{1}{2} \|v\|^2 + C \sum_i w_i \xi_i$$

The resulting optimization in (3.5) is influenced by the weight computed by AdaBoost for each sample.

We evaluated Adaboost with SVM components on the classification standard dataset IRIS (Fisher [1936]) to highlight the genericity of such solution when dealing with multidimensional and multivariated data. Such dataset is composed of 150 samples with 4 attributes ($v_1, v_2, v_3, v_4$) for each sample for describing 3 different types of flowers: Iris Setosa, Iris Versicolour, Iris Virginica. We selected the non-linear classification task of classifying Iris Setosa from the other two. We created 15 multidimensional subspaces composed of combinations of the 4 attributes: $f'_1 = (v_1), f'_2 = (v_2), f'_3 = (v_3), f'_4 = (v_4), f'_5 = (v_3, v_4), f'_6 = (v_2, v_4), f'_7 = (v_2, v_3), f'_8 = (v_1, v_4), f'_9 = (v_1, v_3), f'_{10} = (v_1, v_2), f'_{11} = (v_2, v_3, v_4), f'_{12} = (v_1, v_3, v_4), f'_{13} = (v_1, v_2, v_4), f'_{14} = (v_1, v_2, v_3), f'_{15} = (v_1, v_2, v_3, v_4)$. These subspaces, for each sample, are used as multidimensional features for Adaboost. Thus, we compared the classifier training error by using stump based and SVM based components, see Figure 3.10. Such error shows the subdivision capabilities of the training data of each technique. The graph shows the training error by using Adaboost with SVM components and Adaboost with decision stumps (in blue). Adaboost with SVM obtains a lower error by using 4 times less components than Adaboost with stumps.

It is also relevant to remark the meaning of the parameter $C$ of a soft margin SVM. This parameter tunes the importance given to slack data. The more this value is high, the more it influences the optimization and viceversa. A method to set this value is to use a cross validation method, as suggested by Schölkopf and Smola [2002]. Cross validation is a technique for assessing how the results of a statistical analysis will generalize to an
CHAPTER 3. MULTIMODAL OBJECT DETECTION

Figure 3.10: The graph shows the training error on IRIS dataset (binary non linear classification task: Iris Setosa vs Iris Virginica and Iris Versicolour) by using AdaBoost with SVM components (in red) and AdaBoost with decision stumps (in blue). AdaBoost with SVM obtains a lower error by using 4 times less components than AdaBoost with stumps.

independent data set. In particular in our case we employed a *k-fold cross validation*. The data set is divided into *k* subsets. Each time, one of the *k* subsets is used as the test set and the other *k*−1 subsets are put together to form a training set. Therefore, a time consuming search is run for a discrete set of *C* values that span values (10\(^{-4}\), 10\(^{4}\)) of the real axis. The values for which local minima are found are considered for a finer search in order to select the most suitable value *C*.

In the first work of Spinello and Siegwart [2008a] the features computed on segments have been used to learn an AdaBoost cascade, inspired by the work of Viola and Jones [2002] and Zhu et al. [2006]. AdaBoost cascade reorganizes the Algorithm 1 in a tree of stages. If every stage outputs label 1 in classifying a sample, then a label 1 response is given. If one stage in the tree outputs −1, then a negative response is given. The main advantage of this method is to keep the classification properties of AdaBoost but to sensibly reduce the computation time. This is achieved by automatically placing in the first stages highly discriminative classifiers which avoid further classification stages. Usually each stage is trained to have a small false positive rate value. Details of the
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

Algorithm 2: Train an AdaBoost cascade

Require: \((b_1,l_1) \ldots (b_m,l_m)\) where \(b_i \in \mathcal{B}\) and \(l_i \in (-1,1)\)

1: \(\phi_M\): maximum false positive rate per cascade stage.
2: \(\phi_i\): stage \(i\) false positive rate
3: \(\Phi_M\): target overall false positive rate.
4: \(\Phi\): overall false positive rate.
5: \(\Delta\): overall detection rate.
6: \(\delta_i\): stage \(i\) detection rate.
7: while \(\Phi > \Phi_M\) do
   8: \(i \leftarrow i + 1\)
   9: while \(\phi_i > \phi_M\) do
      10: Select best classifier \(h_t\) on weighted data with error at least \(\epsilon_t < 0.5\)
      11: Update sample data weight as AdaBoost
      12: Evaluate current stage true positives \(\delta_i\) and false positives \(\phi_i\)
   end while
   13: Update \(\Phi = \Phi \phi_i\)
   14: Update \(\Delta = \Delta \delta_i\)
end while

AdaBoost cascade are shortly given in Algorithm 2.

In order to obtain a probabilistic value out of a classification result we adopted the following procedure. Each of the cascaded-SVM classifiers \(h_i\) yields a label \(l_i = (-1,1)\) for a given input feature vector \(b_j\). The overall detection probability can then be formulated as:

\[
p(o \mid b_j) = \frac{\sum \alpha_i \cdot h_i(b_j)}{\sum \alpha_i}
\]

(3.6)

where \(\alpha_i\) are the weak classifiers weight computed in the learning phase of AdaBoost (see line nr. 7 of Algorithm 1) and \(o\) is a binary variable that expresses the existence of an object. We refer in the text to the SVM based AdaBoost classification technique as SBA.

3.6.3.3 Classifying range data with Conditional Random Fields

In (Spinello et al. [2009b]) we moved a step forward in the field of range based detection methods and we addressed the task of classifying multiple categories of objects: cars, pedestrians and background. In order to achieve this goal we needed a more sophisticated and sensitive method. For the detection of objects in 2D laser range scans, several approaches have been presented in the past (see for example Arras et al. [2007], Premebida et al. [2007]). Most of them have the disadvantage that they disregard the conditional dependence between data segments in a close neighborhood. In particular, they can not model the fact that the label \(l_i\) of a given segment \(S_i\) is more likely to be \(l_j\) if we know that \(l_j\) is the label of \(S_j\) given that \(S_j\) and \(S_i\) are neighbors. One way to model this conditional dependency is to use Conditional Random Fields (CRFs) (Lafferty et al. [2001]), as shown by B. Douillard [2008]. CRFs represent the conditional probability \(p(l \mid f^S)\) using an
CHAPTER 3. MULTIMODAL OBJECT DETECTION

An undirected cyclic graph, in which each node is associated with a hidden random variable \( l_i \) and an observation \( f^S_i \). In our case, \( l_i \) is a discrete label that ranges over 3 different classes (pedestrian, car and background) and \( f^S_i \) is a feature vector extracted from the 2D segment \( S_i \) in the laser scan.

Assuming a maximal clique size of 2 for the graph, we can compute the conditional probability of the labels \( l \) given the observations \( S \) as:

\[
p(l | f^S) = \frac{1}{Z(f^S)} \prod_{i=1}^{N} \phi(f^S_i, l_i) \prod_{(i,j) \in E} \psi(f^S_i, f^S_j, l_i, l_j),
\]

where \( Z(f^S) = \sum_{l} \prod_{i=1}^{N} \phi(f^S_i, l'_i) \prod_{(i,j) \in E} \psi(f^S_i, f^S_j, l'_i, l'_j) \) is usually called the partition function and \( E \) is the set of edges in the graph. \( \phi \) and \( \psi \) represent node and edge potentials.

Conditional Random Fields have several advantages with respect to Hidden Markov Models (HMM), Maximum Entropy Markov Models (MEMM) and Markov Random Fields (MRF). HMMs need to enumerate all possible observation sequences, have very strict independence assumptions on the observations and are not practical to represent multiple interacting features or long-range dependencies of the observations. MEMMs, like CRFs, are conditional models that allow arbitrary, non-independent features on the observations and relax strong independence assumptions in generative models. However, MEMMs suffer from the label bias problem: the solution is biased towards states with fewer outgoing transitions. This problem has been solved in CRFs, allowing that some transitions have a stronger influence than others depending on the corresponding observations. A Conditional Random Field can be seen as a special kind of a Markov Random Field. The main difference between these two models is that the node potential (or evidence function of (2.26)) for the \( j \)-node in an MRF is a function of the \( j \)-observation and the \( j \)-state, in CRF this can be relaxed to be a function of all the observation data. More important, the right part of equation (2.26) shows that the edge potential of MRF is independent on the observations but depends just on the states. A state in MRF can influence its neighborhood just indirectly, in CRF this is achieved by explicitly taking into account observations and states in the edge potential.

To determine the node and edge potentials \( \phi \) and \( \psi \) we use the log-linear model:

\[
\phi(f^S_i, l_i) = e^{u_n \cdot f_n(f^S_i, l_i)} \tag{3.8}
\]

\[
\psi(f^S_i, f^S_j, l_i, l_j) = e^{u_e \cdot f_e(f^S_i, f^S_j, l_i, l_j)} \tag{3.9}
\]

where \( f_n \) and \( f_e \) are feature functions for the nodes and the edges in the graph, and \( u_n \) and \( u_e \) are feature weights that are determined in the training phase. The computation of the partition function \( Z \) is intractable due to the exponential number of possible labelings \( l' \). Instead, we compute the pseudo-likelihood, which approximates \( p(l | f^S) \) and is defined by the product of all likelihoods computed on the markov blanket (direct neighbors) of node \( i \).
Here, \( N(f_S^i) \) denotes the set of direct neighbors of node \( i \). In the training phase, we compute the weights \( u = (u_n, u_e) \) that minimize the negative log pseudo-likelihood together with a Gaussian shrinkage prior as in Ramos et al. [2007]:

\[
L(u) = - \log p(l | f^S) + \frac{(u - \hat{u})^T(u - \hat{u})}{2\sigma^2}
\]  

For the minimization of \( L \), we use the L-BFGS gradient descent method (Liu and Nocedal [1989]). Once the weights are obtained, they are used in the inference phase to find the labels \( l \) that maximize equation (3.7). Here, we do not need to compute the partition function \( Z \), as it is not dependent on \( l \). We use max-product loopy belief propagation (BP) to find the distributions of each label \( l_i \). The final labels are then obtained as those that are most likely for each node.

In our case the Delaunay triangulation among segments defines the lattice of the network (see Section 3.6.1.2). We use a set of statistical and geometrical features, already explained in Section 3.6.2, for the nodes of the CRF. However, we do not use these features directly in the CRF, because, as stated in Ramos et al. [2007] and also from our own observation, the CRF is not able to handle non-linear relations between the observations and the labels. Instead, we apply AdaBoost (Freund and Schapire [1997]) to the node features and use the outcome as features for the CRF. For our particular classification problem with multiple classes, we train one binary AdaBoost classifier for each class against the others. As a result, we obtain for each class \( c \) a set of \( t \) weak classifiers \( h_i \) (in this case decision stumps) and corresponding weight coefficients \( \alpha_i \) so that the sum

\[
g_c(f_S^i) := \sum_{i=1}^{t} \alpha_i^c l_i^c(f_S^i)
\]  

is positive for observations assigned with the class label \( c \) and negative otherwise. It is important to remark that equation (3.12) corresponds to the argument of the sign() function of line nr. 11 of Algorithm 1. Note that the absolute value of \( g_c \) can be interpreted as a classification quality. To obtain a classification likelihood, we apply the logistic function \( a(x) = (1 + e^{-x})^{-1} \) to \( g_c \). We do this for two reasons: first the resulting values are between 0 and 1 and can be interpreted as likelihoods of corresponding to class \( c \). Second, by applying the same technique also for the edge features, the resulting potentials are better comparable. Thus, the node feature function \( f_n \) of the segment features \( f_S^i \) and the label \( l_i \) is computed as:

\[
f_n(f_S^i, l_i) = a(g_{l_i}(f_S^i))
\]
For the edge features $f_e$ we compute two values, namely the Euclidean distance between the centroids $c_i$ and $c_j$ of the segments $S_i$ and $S_j$, along with a value $g_{ij}$ defined as:

$$g_{ij}(f_S^i, f_S^j) = \text{sign} \left( g_i(f_S^i) g_j(f_S^j) \right) \cdot (\|g_i(f_S^i)\| + \|g_j(f_S^j)\|)$$  \hspace{1cm} (3.14)

Thus, the value of $g_{ij}$ has a positive sign if AdaBoost classifies $f_S^i$ and $f_S^j$ into the same class and otherwise it is negative. The absolute value of $g_{ij}$ is the sum of the classification qualities of AdaBoost. If $g_i(f_S^i)$ and $g_j(f_S^j)$ are far from 0 then $g_{ij}$ has a high value, and vice versa. To summarize, we define the edge features as:

$$f_e(f_S^i, f_S^j, l_i, l_j) = \begin{cases} 
  (a(\|c_i - c_j\|), a(g_{ij}(f_S^i, f_S^j)))^T & \text{if } l_i = l_j \\
  (0, 0)^T & \text{otherwise}
\end{cases}$$  \hspace{1cm} (3.15)

The intuition behind equation (3.15) is that edges that connect segments with equal labels have a non-zero feature value and thus yield a higher potential. The latter is sometimes referred to as the generalized Potts model (see Anguelov et al. [2005], Potts [1952]).

### 3.6.4 Training and application of Range Classifiers

In this section we clarify and summarize the usage of the proposed techniques for detecting objects in range data. Two are the detection methods that have been explained: detection by using SBA (see Section 3.6.3.2), detection by using CRF (see Section 3.6.3.3).

#### Training a SBA classifier

A set of labeled range data laser scans is provided for training the detector. In this set a class label is assigned for each segment of each laser scan. The procedure can be summarized as:

- Each labeled scan is clustered by using JDC on connected Delaunay segments (see Section 3.6.1.2).
- Range data based features are computed for each cluster in each scan (see Section 3.6.2)
- Training of a SBA classifier is run by using features and associated class labels of each cluster.
- Weights and weak classifiers parameters are stored.

#### Detection with a SBA classifier

The detection procedure of objects in unknown range data can be summarized as:

- A laser scan is received.
- The range data is clustered by using JDC on connected Delaunay segments (see Section 3.6.1.2).
3.6. STRUCTURE INFORMATION FROM LASER DATA ANALYSIS

- Range data based features are computed for each cluster (see Section 3.6.2).
- The SBA trained classifier processes the features associated to each cluster and it produces, for each cluster, a detection likelihood (see equation (3.5)).

**Training a CRF classifier** A set of labeled range data laser scans is provided for training the detector. In this set a class label is assigned for each segment of each laser scan. The procedure can be summarized as:

- Each labeled scan is clustered by using JDC segmentation (see Section 3.6.1.1).
- Range data based features are computed for each cluster in each scan (see Section 3.6.2)
- Training of an AdaBoost classifier is run by using features and associated class labels of each cluster (see Section 3.6.3.1).
- Weights and weak classifiers parameters are stored.
- CRF is trained by using the same set of labeled scan data by using node features and edge features. Node features depend on the output of the previously trained AdaBoost classifier, see equation 3.13. The graph for the CRF is defined by using a Delaunay triangulation among segments
- CRF node and feature weights are stored.

**Detection with a CRF classifier** The detection procedure of objects in unknown range data can be summarized as:

- A laser scan is received.
- The range data is clustered by using JDC segmentation (see Section 3.6.1.1).
- Range data based features are computed for each cluster (see Section 3.6.2).
- The trained AdaBoost classifier is run on the features associated to each cluster (see Section 3.6.3.1).
- The graph of the CRF is defined by using a Delaunay triangulation among segments. Node features and edge features are computed. Note that node features depend on the output of the AdaBoost classifier, see equation 3.13. Belief propagation is finally run on the trained CRF in order to obtain a detection likelihood for each segment of the line scan.
CHAPTER 3. MULTIMODAL OBJECT DETECTION

3.7 Experimental evaluations of range-based data detection

In this section we provide experimental evaluations of the methods regarding range data classification. Data logs have been taken with our urban mobile robotic platform. Experiments have been conducted by using the sensor setup configurations presented in the Appendix section A.3.

3.7.1 Quantitative measures of performance

We use several standard measures of classification performance, principally: precision, recall, receiver operating characteristics (ROC), Equal Error Rate (EER), Area under the ROC curve (AUC). The **precision** and **recall** values are defined as:

\[
\text{Precision} = \frac{tp}{tp + fp} \quad \text{Recall} = \frac{tp}{tp + fn}
\]  

where \( tp \) is the number of true positives, \( fp \) is the number of false positives and \( fn \) is the number of false negatives. The ROC is represented by plotting true positive rate vs false positive rate. AUC, for an ROC curve, is equal to the probability that a classifier will rank a randomly chosen sample instance higher than a randomly chosen negative one. Equal Error Rate is a value evaluated in a precision-recall plot to assess the quality of a classifier: it is the point in the curve when the value ‘precision’ is equal to the value ‘recall’. If EER is equal to 1, a perfect classification is obtained.

3.7.2 Real World Datasets: Laser data

We have collected several datasets for evaluating the different proposed techniques.

**Urban Dataset I - Laser (UD1):** Laser and camera datasets have been retrieved in two different outdoor scenarios: a parking lot and a university campus dataset. The parking lot dataset consists in a staged "road like" scenario: the car is moving around a parking lot, meanwhile people crosses the road, cars are parked on the side of the road and road intersections are present. Pedestrians are wearing backpacks, hats and jackets to increase the appearance variability content in order to test the generalization of the vision classifier. The university campus dataset presents a challenging and cluttered environment with a quantity of walking pedestrians with different shapes, speeds and distribution in space. The car is parked in front of a university building. Here posts and vertical structures challenge both the laser and the vision algorithm components. The parking lot data set is composed of 626 scans, the university campus dataset of 264 scans. Laser angular resolution has been set to 0.50 degrees.

**Urban Dataset II - Laser (UD2):** We evaluated our technique on a challenging urban scenario dataset. We produced two sets in different moments of the year with different weather and traffic conditions. Laser angular resolution has been set to 0.25 degrees in order to retrieve high resolution data in both sets. The first set consists in a collection of 2 sequences 482 scans and 182 scans. Data is collected around the city of Zürich in Switzerland. The first set (UD2.1) consist in selected sequences in which pedestrians are
walking, crossing, standing and where severe occlusions are present. The second (UD2.2) has been collected by driving our mobile platform in a loop of approximately 1km length for retrieving cars and pedestrians in a real busy urban environment. This set consists in a collection of 1675 laser scans and 600 laser scans used just for training the range classifier.

In both datasets laser scans have been manually labeled by using associated image frames as reference for the ground truth. Labeling is obtained by manually selecting and assigning a class label to the segments in the range data. A suite of MATLAB scripts have been used to simplify this process.

### 3.7.3 Evaluating of SBA classification for Pedestrian Detection

We here evaluate the quality of object detection in range data by using the SBA detection technique, summarized in Section 3.6.4.

**Training** We are interested in a binary classification problem: our aim is to learn a classifier that is able to robustly distinguish a pedestrian from all the rest. Thus, a training set is produced and clusters in range data are manually labeled as pedestrian or background. We evaluate the algorithm by using the dataset UD1. The parking lot data set is composed by 1136 positive and 7008 negative clusters. The campus dataset is composed by 498 positive and 3598 negative clusters.

Each of the presented dataset is divided in a training set and in a test set. The campus training set is composed by a 50% random choice of positive samples and negative samples, the same sampling method is applied to the parking lot dataset. The testing sets are composed by the remaining clusters. The range features classifier training dataset is the union of both training datasets. See Table 3.2.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Positive</th>
<th>Negative</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parking Lot</td>
<td>568</td>
<td>3504</td>
<td>4072</td>
</tr>
<tr>
<td>Uni Campus</td>
<td>249</td>
<td>1799</td>
<td>2048</td>
</tr>
<tr>
<td>Total</td>
<td>817</td>
<td>5303</td>
<td>6120</td>
</tr>
</tbody>
</table>

**Table 3.2:** Training set for laser classifier.

The resulting AdaBoost SVM cascade (see Section 3.6.3.1) is composed by 4 stages of 18 features each. It is interesting to analyze which features have been repeatedly selected by AdaBoost in the first stage of the cascade. This gives an information about the descriptivity quality of certain range features. Features repeated more than once in the first stage of the cascade are showed in Table 3.3.

The resulting selected features are a balanced combination of shape description and points distribution statistics often related to the distance of the laser segment from the origin. Moreover, we studied the segments distance separation as a cue for detecting pedestrians in outdoor scenario. Therefore, we added this value as \( f_{d_2}(S_i) \) to the feature set \( F \), computed as the distance of the nearest neighbor of the segment \( S_i \). The automatic
Table 3.3: Features repeated more than once in the first stage of the cascade for the laser classifier. They are a mixture of 1D and multidimensional features.

The process of building the boosted cascade excluded this features due to a lower performance with respect to the other features. This results differ from the work of Arras et al. [2007] mainly due to a very different training set (outdoor vs indoor), a different segmentation method (Delaunay JDC vs JDC) and a richer feature set.

Detection In this section we evaluate the quantitative performance of the laser classifier. In Table 3.4 we show the confusion matrix of the range data classifier for the parking lot testing set UD1.

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Prediction</th>
<th>Tot</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>517 (91.1%)</td>
<td>51 (8.9%)</td>
</tr>
<tr>
<td>N</td>
<td>351 (10.0%)</td>
<td>3153 (90.0%)</td>
</tr>
</tbody>
</table>

Table 3.4: Confusion matrix for parking lot dataset - (LASER)

The detection rate for is very high (over 90%) and false positive rate/false negative rate is low (less 10% for both). Even though the environment resembles a road scenario, the persons appearing in range data are well separated from the background, the background is mostly composed by a collection of straight segments and a small amount of clutter data is present. This ease the task of the classifier that has to evaluate data with low classification ambiguity.

The second part of UD1 is based on the cluttered environment of a university campus. Results are given in Table 3.5.

The detection rate is around 65% for the laser classifier. This result can be explained by the complexity of the environment and, moreover, by the clutter present in this scenario:
3.7. RANGE-BASED DATA DETECTION EVALUATION

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Prediction</th>
<th>Tot</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>161 (64.7%)</td>
<td>88 (35.3%)</td>
</tr>
<tr>
<td>N</td>
<td>536 (30.0%)</td>
<td>1273 (70.0%)</td>
</tr>
</tbody>
</table>

Table 3.5: Confusion matrix for university campus dataset - (LASER)

![ROC curve laser classifier](image)

**Figure 3.11:** ROC curve for the laser classifier evaluated on dataset UD2. AUC is $\approx 0.8$.

multiple pedestrians in a small space change the retrieved cluster shape and point distribution in range data. In such case, points belonging to several pedestrians can be clustered together or occlusions can fragment a single segment in several smaller ones. This segmentation problem affects range feature values, producing ambiguity in the classification task.

In order to assess the generalizing ability of this classifier we run an evaluation on the first set of dataset UD2.1. The laser detector is evaluated by showing the ROC curve in Figure 3.11. Area Under the Curve (AUC) is 0.78. It is interesting to see how important is the data angular resolution for classifying pedestrians. This dataset is similar, if not more complex than then UD1-university campus dataset but angular resolution is double. On the latter the performances are better due to this reason.

### 3.7.4 Evaluating of CRF classification for Multiclass Detection

We here evaluate the quality of object detection in range data by using the CRF detection technique, summarized in Section 3.6.4. With this method we are interested in detecting
multiple categories of objects: cars and pedestrians. Conditional Random Fields have been used to detect objects classes from laser data.

**Training**  Our laser training data consists of 600 annotated scans with pedestrians, cars and background taken from the laser training dataset UD2.2. There is no distinction of car views in the laser data as the variation in shape is low. As a first step AdaBoost classifier of range features is trained on this set. Then we use the trained AdaBoost classifier output to produce node features values for CRF. Then, CRF is trained in order to set the node and edge features weights.

**Detection**  We evaluated the performance of the laser based detector for pedestrian detection in Figure 3.12-left. There, we visualized a comparison between the proposed Boosted CRF and the simple AdaBoost classification of JDC segments (AJDC) in order to visualize the introduced performance enhancements. AJDC classifies JDC segments regardless of the neighborhood state and it can be seen as a flavor of an SBA classification method. We therefore compared the performance improvements introduced with CRF with respect to the simpler AdaBoost based method. It is interesting to notice that taking into account the segments neighborhood in CRF plays an important role in the ability to increase detection rate and reduce the number of false positives, the AJDC curve is always below the CRF one and it decreases earlier than the CRF curve. EER is 64.23% for CRF and 57.09% for AJDC.

\[ \text{Figure 3.12: Quantitative evaluation (precision-recall graphs) for multiclass object detection by using laser range data. We show a comparison between Boosted CRF and AdaBoost classification of JDC segments (AJDC) in order to visualize the introduced performance enhancements. Evaluation of pedestrian category is shown on left figure and car category is shown on right figure. It is important to notice that our proposed method outperforms AJDC for both object categories.} \]

We then evaluated the performance of our system in case of car detection. Laser is compared with AJDC in Figure 3.12-right and also in this case CRF has better results
with respect to AJDC. EER is 74.89% for CRF and 70.57% for AJDC. It is interesting to notice that cars are in general easier to detect with respect to pedestrians. Intuitively, cars are rigid objects with much less geometrical variability than flexible and highly complex pedestrians.
CHAPTER 3. MULTIMODAL OBJECT DETECTION

3.8 Appearance Information from Image Data Analysis

We here describe the approaches taken for detecting pedestrians and cars from camera images. We based our work on state of the art techniques like Histogram of Oriented Gradients (HOG) detectors, introduced by (Dalal and Triggs [2005]) or Implicit Shape Model detectors (ISM), by (Leibe and Schiele [2004]). In the first paper (Spinello and Siegwart [2008a]) we detected pedestrian by using HOG, in the remaining works we progressively produced extensions of ISM, called ISMe (Spinello et al. [2008a,b, 2009b]). Our extensions focus on feature selection, matching and weighting, hypotheses voting, subparts reasoning and multiclass capabilities.

3.8.1 Terminology: interest points and descriptors

Before introducing the vision based object detection methods, it is convenient to clarify some terminology used in this chapter. Very often there is confusion in the definition between the term interest points and descriptors for images. In our works we intend interest points as the result of an interest point detector algorithm: a well founded definition for identifying a local area in the image, rich in terms of local information, that is stable in space and, possibly, in scale. An interest point is expressed by the coordinates of the center of a circular (or ellipsoidal) area and its radius (also called scale). A region descriptor is a way of encoding the content of a local image area in a compact form. It describes the region located by the interest point detector as a d-dimensional vector. Most of the region descriptors are compared by using Euclidean distance.

3.8.2 Classifying Pedestrians with Histogram of Oriented Gradients

Edge orientation histograms descriptors have been successfully used for object detection and recognition (Belongie et al. [2002], Freeman and Roth [1994], Lowe [2003], Mikolajczyk et al. [2004]). The concept of classification of dense and local histograms of oriented gradients (HOG) for pedestrian detection has been introduced by Dalal and Triggs [2005]. The aim of such technique is to describe an image by a set of local histograms. These histograms collect occurrences of gradient orientation in local parts of the image.

In order to compute HOG descriptors three steps have to be processed:

- Computation of the image gradient.
- Subdivision of the image in cells.
- Computation of an histogram of orientation for each cell.
- Histogram normalization by using blocks.

The image gradient is computed by using two convolutions on the image. The first, that detects horizontal component, is processed by using a kernel \((-1 0 1)\); the second, for the vertical component, uses a \((-1 0 1)^T\) kernel. The gradient can be signed or
unsigned, we used an unsigned gradient: a white object placed on a black background would have same gradient image of a black object on a white background.

In the original method of Dalal and Triggs [2005] a rectangle window of fixed size defines the object. This window is divided into overlapping squared cells of fairly small size ($8 \times 8$ pixels). This cell size helps to capture local features in the detection window but, on the other hand, does not explicitly consider bigger size shape relations. To overcome this limitation and to accelerate the detection process, we have implemented the method proposed by Zhu et al. [2006]. Cells in different locations and with aspect ratios, are added to enrich the feature set and to capture more information. For each cell, a local histogram is composed by accumulating gradients into bins for each orientation. Histogram votes are weighted by the magnitude of the gradient in order to take into account the strength of the visual responses for a given angle. Intuitively, a pixel of an edge would have a higher weight than a point in a uniformly colored region. In case of using an unsigned image gradient, only orientations defined by values in the closed interval $[0, \pi]$ are collected. A very important parameter to set is the number of bins. The larger the number of bins, the more detailed the histogram is. This value has been set experimentally to 9 bins after conducting an extensive set of experiments (Dalal and Triggs [2005]).

It is necessary to normalize cell histograms due to the light variability in the images. Cell histograms are normalized according to the values of the neighboring cell histograms. The normalization is done among a $2 \times 2$ group of cells, which is called a block. In a block the 4 histograms are concatenated together to form a 36-binned histogram that is normalized to an $L_1$ (Manhattan distance) unit length.

We designed the ratio between block width and block height to be any of the following ratios $(1 : 1)$, $(1 : 2)$ and $(2 : 1)$. Moreover we consider all blocks whose size ranges from $12 \times 12$ to $64 \times 128$ using an increasing step of $\{4, 6, 8\}$ pixels. In our implementation 5245 HOG descriptors $\mathcal{F}_{\text{HOG}}$ are computed in each detection window. Note that according to design of each block, an histogram from a given cell can be involved in several block normalizations. Therefore we redundantly encode the information of overlapping cells in different normalizations. Porikli [2005] introduced the Integral Histogram technique to efficiently compute histograms over arbitrary rectangular image regions. Inspired by this work it is possible to compute an HOG feature very efficiently. We discretize each pixel’s gradient orientation into 9 angles and store an integral image for each angle. Therefore, we can collect cell histograms by just using the stored discretized integral images and compute efficiently HOG descriptors for each block.

For the classification of these descriptors we employed an AdaBoost cascade of linear soft margin SVMs (see Algorithm 2). AdaBoost automatically selects, orders and weights the most descriptive features of the big feature set. In order to shorten the training process time, for each training round of AdaBoost we selected 5% of the total descriptors, as suggested in Zhu et al. [2006].

The detection stage works as the one suggested in the paper of Viola and Jones [2002]. The detection window is scrolled through the image at different scales. If all the stages of the AdaBoost cascade are passed, an object detection is marked for that area. Then, a non-maxima suppression is run in order to remove detections that are too close together.
A detection probability is computed as the one shown in formula (3.6).

3.8.3 Classifying Pedestrians with Extended Implicit Shape Models

We here present our other image-based object detector that is mostly inspired by the work of Leibe and Schiele [2004] and Leibe et al. [2005] on scale-invariant Implicit Shape Models (ISM). An ISM is a generative model for object detection and has been applied to a variety of object categories including cars, motorbikes, animals and pedestrians. In this work, we extend this approach and we explain the steps for learning an object model in the original ISM framework and our extensions.

An Implicit Shape model consists of a codebook $I$ of local appearances and of a spatial probability distribution $V$ which specifies where each codebook entry may be found on the object. The spatial vote distribution has the following properties: it is defined independently for each codebook entry and it is estimated in a non-parametric manner for each codebook entry. The first property allows to combine object parts during recognition that were initially observed on different training examples. The second property enables to model non Gaussian distributions of codebook entries in great detail.

3.8.3.1 Generating a codebook

A codebook $I$ of local appearances of a particular object category is described by a set of (scale invariant) region descriptors computed around standard interest points. This has been inspired by the work of Agarwal and Roth [2002]. Literature standard region descriptors (e.g: SIFT (Leibe et al. [2005]), Shape Context (Belongie et al. [2002]), GLOH (Mikolajczyk and Schmid [2005])) and interest point detectors (e.g. Harris-Laplace, Hessian-Laplace (Mikolajczyk and Schmid [2005])) can be used. Thus, it is important to remark that ISM is not restricted to a single kind of descriptor: several different descriptors can be used for the purpose of describing an object.

A training set is needed in order to create a codebook. The training set is constituted by a collection of labeled images that represent an object category. The labeling is expressed by a binary segmentation mask associated to each training image, that defines the ‘inside’ and the ‘outside’ of an object. Local region descriptors are extracted from interest points inside the objects’ labeling and collected in a set $C$. We call $O^+$ the interest points found inside the labeling and $O^−$ the ones in the background. The set $C$ usually contains thousands of elements. For practical reasons we need to reduce the set cardinality by using a clustering method. Due to the fact that the elements of $C$ lie in a high dimensional space, agglomerative clustering has been favored as an effective and practical solution. Precisely, agglomerative clustering with average linkage is used in order to maintain compactness in the descriptor space, that is often high dimensional (128 in case of SIFT, 64 in GLOH, 36 in case of Shape Context descriptors). In order to use this method of clustering we need to define a similarity measure that is used to discriminate when two data points are considered to be in the same group. At this regard, Euclidean distance between two descriptors has been used. An algorithm overview of agglomerative clustering is given in Algorithm 3. We use a distance threshold $τ_I$ to group data. At the end of the clustering
Algorithm 3 Agglomerative Clustering with Average Linkage

Require: $c_i \in C$

Require: All the data is initialized as a cluster center $c^k_i$ in $\mathcal{K}_i$.

1: $D$ is a similarity function
2: while $\tau > \tau_{\text{min}}$ do
3: $\tau_{\text{min}} \leftarrow \text{argmin}(D(c^k_i, c^k_j))$ \{Take the data points with smallest similarity distance.\}
4: Tag $c^k_i$ and $c^k_j$ as neighbors in the same cluster $\mathcal{K}_v$
5: Recompute centroid of $\mathcal{K}_v$
6: end while

algorithm only the cluster centers are stored as entries of the codebook $\mathcal{I}$.

The second step of Implicit Shape Model generation for certain object category is to create, from a codebook of local appearances $\mathcal{I}$, a spatial probability distribution $\mathcal{V}$ related to each codebook entry. Codebook entries $\mathcal{I}$ are matched with the descriptors found in all the images of the training set. In such process we consider an entry matched if it obtains a Euclidean distance less than $\tau_T$ (the same threshold used during clustering) from an image feature descriptor. For each matched entry, we store the relative position of the associated interest point with respect to the object center. The set of relative displacements for all the codebook entries, also called votes, defines the distribution $\mathcal{V}$. The process of building a codebook is visually explained in Figure 3.13. Intuitively, ISM can be seen as a meta-detector, a smart way of describing with a bag-of-features approach the features distribution (in descriptor and image space) of an object category.

3.8.3.2 ISM Extension: Generating a Superfeatures Codebook

The generation process of a codebook for ISM does not contain any feature selection method. This has two potential disadvantages: first, a codebook for an object category may contain many entries and second, each entry may cast a big quantity of votes. It can be argued that by increasing the cluster distance $\tau_I$, during the process of generating the codebook, the number of codebook entries can be effectively reduced. That way, the variance represented by each clustered entry in the high dimensional descriptor space can be big, therefore the possibility of matching with visually different descriptors can significantly increase. Moreover, a codebook is usually matched to descriptors found in an image by using the same distance $\tau_T$ used for generating the codebook. Therefore, each codebook entry would be associated to a big quantity of votes due to the wider matching radius and due to the feature mismatches in the creation of the codebook. This latter disadvantage is present even if an occurrence of the object is not likely given the training data, and it might cause many false positive detections.

The goal of a superfeature codebook is to overcome these disadvantages by retrieving only selected descriptors that may cast strong votes. We define superfeatures as features that are stable in image space and in descriptor space. This means that a superfeature is frequently found, in the training set, in approximately the same position with respect
Figure 3.13: Codebook generation for ISM. A training set is composed of a collection of images of an object category and labeled binary images that define the ‘inside’ and the ‘outside’ of each object (top). Only features descriptors of the object interior are considered for clustering (in red). The centers of the clustered descriptors compose the codebook. These codebook entries are then matched again with the images from the training set (bottom). The relative position of each feature match is associated to the activated entry to compose an Implicit Shape Model.
to the object center and its position is stable also in descriptor space. This definition ensures that for superfeatures a high evidence of the occurrence of the object is combined with a high probability to encounter an interest point. In order to reach both of these goals we introduce a two-step process. The superfeature set is computed from the training data. First, all the interest points $O^+$ from the training image set, found inside the area indicated by the segmentations that define the objects, are accumulated in a continuous three dimensional cartesian space. The three dimensions are: relative displacement of the interest point with respect to the center of the object ($\Delta x, \Delta y$) and scale $s$ in which the interest point has been detected. In this space we are interested in high density areas, loci where interest points are very often found for a certain object category. Therefore, a mean shift mode estimation (Comaniciu et al. [2001]) with uniform three dimensional spherical kernel is run in order to locate maxima of local density $v^*_i$. Mean-shift is a non-parametric space analysis technique: it is a procedure for locating stationary points of a density function given discrete data sampled from that function. In our case, it is useful for detecting the modes of the interest point distribution. The second step

Figure 3.14: Superfeatures codebook generation. Superfeatures are stable features in image and descriptor space. Interest points are accumulated in a continuous space from the training images. Mean shift mode estimation is run in order to locate high density areas. There, an agglomerative clustering algorithm is run on associated set of descriptors in order to select 50% of most populated clusters. This figure shows Shape Context descriptors at Hessian interest points (in red) for the class ‘pedestrian’. The position of the superfeatures are depicted in green.
consists in clustering the region descriptors associated to the interest points found in the kernel radius of each \( v_i \). Segmentation in this high-dimensional space is obtained by using agglomerative clustering with average linkage. In order to extract just the most important content we select the 50% of the cluster centers that correspond to the most populated clusters in descriptor space. These clustered entries define the superfeature codebook entries \( I^* \); their associated collection of \( v^* \) defines the set of codebook votes \( V^* \). The resulting superfeatures codebook is smaller than the standard ISM codebook and each entry is associated to less votes. Figure 3.14 shows a visual explanation of the superfeature codebook generation.

It is interesting to notice that the superfeatures inherently reflect the skeleton of the object. In case of a pedestrian, superfeatures are mostly taken in the V-shaped area between the legs, and nearby the shoulders. Even though this result is strictly related to the kind of interest point detector (Harris and Hessian interest points are located on corner/blobs) it intuitively reflects distinctive local areas for detecting pedestrians and it has similarities with several scientific results (e.g. see the discussion of Dalal and Triggs [2005] on the high classification weight that such areas receive).

### 3.8.3.3 Generating object hypotheses

In this section we explain how to generate object detection hypotheses with ISM by using the learned codebook. The intuition behind ISM is that each descriptor found in an image is matched with the codebook (or superfeatures codebook). Activated codebook entries vote for possible object centers in a continuous voting space. Local maxima in such space define object hypotheses.

We can formalize the problem by using probabilities. An Implicit Shape model consists of a codebook \( I \) and a set of votes \( V \). The \( K \) elements of \( I \) are local region descriptors \( d^c_1, \ldots, d^c_K \) and \( V \) contains for each \( d^c_i \) a set of \( D_i \) local displacements \( \{(\Delta x_{ij}, \Delta y_{ij})\} \) and scale factors \( \{s_{ij}\} \) with \( j = 1, \ldots, D_i \). The interpretation of the votes is that each descriptor \( d^c_i \) can be found at different positions inside an object and at different scales. To account for this, a vote is cast from the interest point associated to the descriptor that matches \( d^c_i \) to the center of the object, as it is found in the labeled training data set. We can think of this as a sample-based representation of a spatial distribution \( p(o, \hat{x} | d^c_i, x_i) \) for each \( d^c_i \) matched at a given image location and scale \( x_i = (x_i, y_i, s_i) \) where \( o \) is a binary variable indicating the presence of an object and \( \hat{x} = (\hat{x}, \hat{y}, \hat{s}) \) denotes the estimated center and scale of the object.

In the detection phase, we compute interest points \( x'_I \) and corresponding region descriptors \( d'_I \) at various scales on a given test image \( I \). The descriptors in the image are matched to the codebook entries by using Euclidean distance: all the distances less than \( \tau_I \) are considered a match. Thus, a matching probability is computed for each matched codebook entry:

\[
p(d^c_i | d'_I) = \frac{\|d^c_i - d'_I\|}{\tau_I}
\] (3.17)
The likelihood to detect an object from a single matched descriptor $d^I_J$ is:

$$
p(o, \bar{X}_i | x^I_J, d^I_J) = \left(p(o, \bar{X}_1 | x^I_J, d^I_1), \ldots, p(o, \bar{X}_K | x^I_J, d^I_K)\right)$$  \hspace{1cm} (3.18)

$$
p(o, \bar{X}_i | x^I_J, d^I_J) = p(o, \bar{X}_i | d^C_i, x^I_J)p(d^C_i | d^I_J)$$  \hspace{1cm} (3.19)

where $\bar{X}_i$ represents the set of possible object centers and $K$ is the number of matched codebook entries. Equation (3.19) defines the weight of the vote that is cast by each descriptor $d^I_J$ found at location $x^I_J$ for a particular occurrence of objects at locations $\bar{X}_i$. To clarify, if the image descriptor $d^I_J$ matches $K$ codebook entries and each activated codebook entry $d^C_i$ is associated to $V_i$ displacements, then $V_i$ votes for object centers are casted in different directions each with a weight value:

$$
p(o, \bar{x}_k | d^C_i, x^I_J) = \frac{1}{K} \frac{1}{V_i} \bar{x}_k \in \bar{X}_i$$  \hspace{1cm} (3.20)

These votes are casted from the interest point relative to descriptor $d^I_J$ by using the directions associated to each activated codebook entry. In order to know the position of the object center $\bar{x}_k$ originated from the vote casted from the matched codebook descriptor $d^C_i$ with the image descriptor $d^I_J$ we compute:

$$
\bar{s}_k = \frac{s_{d^I_J}}{s_{d^C_i}}$$  \hspace{1cm} (3.21)

$$
\bar{x}_k = x_{d^I_J} - \bar{s} \Delta x_{d^C_i,k}$$  \hspace{1cm} (3.22)

$$
\bar{y}_k = y_{d^I_J} - \bar{s} \Delta y_{d^C_i,k}$$  \hspace{1cm} (3.23)

$$
\bar{s}_k = (\bar{x}_k, \bar{y}_k, \bar{s}_k)$$  \hspace{1cm} (3.24)

where $s_{d^I_J}$ is the scale of the image descriptor, $s_{d^C_i}$ is the scale of the matched entry of the ISM codebook and $(\Delta x_{d^C_i,k}, \Delta y_{d^C_i,k})$ is the $k$-vote direction associated to $d^C_i$.

Thus, our aim is to find locations in which high quantity of votes for object centers converge. The overall detection likelihood in one of such location $\hat{x}$ is then the sum over all votes probability:

$$
p(o, \hat{x} | g^I) = \sum_{\bar{x}_k} p(o, \bar{x}_k | x^I_J, d^I_J)$$  \hspace{1cm} (3.25)

where $\bar{x}_k \approx \hat{x}$ defines all the votes casted nearby $\hat{x}$ and $g^I = (x^I_1, \ldots, x^I_M, d^I_1, \ldots, d^I_M)$. With the sample-based representation, we can find the $\hat{x}$ that maximizes (3.25) by a maxima search.

The voting space, that resembles a probabilistic version of the Generic Hough Transform, is a three dimensional continuous (position and scale) space. We make use of a mean-shift search strategy to assess the exact object location. It is important to highlight that the non-parametric nature of the approach overcomes the usual Gaussian assumption.
Figure 3.15: A mean shift balloon estimator is used to find object centers in position and scale. Ellipsoid changes aspect ratio with respect to the scale: its axes on $x, y$ differently resize with respect to the scale axis due to the intuition that the bigger the scale ratio $s_{d_i}/s_{d_c}$ of a certain vote, the bigger the relative error produced in $x, y$.

Precisely, the local maxima search in the continuous voting space is obtained through a mean shift with a variable kernel, as suggested in Leibe and Schiele [2004]. The idea is that the bigger the scale ratio $s_{d_i}/s_{d_c}$ of a certain vote, the bigger the relative error produced in (3.21),(3.22),(3.23). Therefore, we employed a variable bandwidth mean shift kernel, defined by an ellipsoid that resizes the semi-axes with respect to the scale; it is also called mean shift balloon estimator. The mean shift kernel ellipse $E$ is defined by the axes:

$$E_z = s \gamma_1$$  \hspace{1cm} (3.26)
$$E_x = s e_b$$  \hspace{1cm} (3.27)
$$E_y = s e_b$$  \hspace{1cm} (3.28)

where $\gamma_1$ is the gain factor related to the $z$ semi-axis of the ellipse and $e_b$ is a value that has been fixed to 10% of the object width. The object width is defined by averaging all the objects width in the training data (at scale 1). See Figure 3.15 for a visual explanation of the variable kernel.

In order to accelerate the process of mean shift convergency, a preliminary three dimensional histogram has been built. Starting positions for mean shift would be all that cells that are a local maxima in a $3 \times 3 \times 3$ neighborhood. This decreases dramatically the computation time by avoiding processing uniform or empty areas.

The convergency points, local maxima of votes density, define the object hypotheses, that is where an object could be found in the image.
### 3.8.3.4 ISM Extension: Generating object hypotheses with Superfeatures Codebook

The process of generating detection hypotheses is totally analogous to the one explained in the previous section but the codebook matching is done by using the *superfeatures* codebook $I^*$. Equation (3.20) becomes:

$$p(o, \bar{x}_k | d^C_i, x^I_j) = w^* \frac{1}{K V_i} \bar{x}_k \in \bar{X}_i$$ (3.29)

$w^*$ is a scalar that expresses a confidence gain given to the superfeatures. To clarify: the superfeatures codebook casts votes in the same voting space and in the same way of the normal ISM codebook but each vote has more weight. In a visually simple scenario it is possible to use only the superfeature codebook for detecting objects. In the work of Spinello et al. [2009b], $w^*$ has been set experimentally to 2.0.

### 3.8.3.5 ISM Extension: Weighting the codebook entries and the shape flexibility

Another improvement, introduced in (Spinello et al. [2008b]) before the introduction of the superfeatures codebook and precursor of such idea, has been developed in the scope of feature weighting. The method introduces two kinds of features weights. In the first, object codebook entries are weighted considering their neighborhood in descriptor space. In the second, the idea is to consider the object shape flexibility as a way of defining stable features and to consider the votes coming from those stable features with increased confidence.

Features found in the object silhouette $O^+$ and features found in the background $O^-$ are both collected during the training phase. Therefore, a neighborhood analysis of each $d^O_i$ descriptor is computed considering the quantity of $||O^-||$ background samples in a radius of distance $\tau_I$ (the same value used during codebook generation). This value, called $w'_i$, is then normalized with respect to the cardinality of the entire negative set $D^O$:

$$w'_i = 1 - \frac{||N(d^O_i, \tau_I, D^O)||}{||D^O||}$$ (3.30)

where $N(a, b, C)$ corresponds to the set of neighbors, in descriptor space, of feature $a$ from the set $C$, in a radius with Euclidean distance $b$. This weight gives an information about the distinctiveness of each feature, assigning very low values to positive samples in loci where a high number of negative descriptors are found. We assume that the features contained in $O^-$ are a good representation of the object background. This method decreases the amount of false positive matching and it can be seen as a soft way of expressing a $k$-nearest neighbor classification.

The other proposed improvement in vote weighting is to consider the vote distribution of $V$ associated the codebook entries $I$ as an informative cue of the object pose. The votes of the codebook are analyzed for positional stability with respect to the object center: the more the same codebook entry $i$ is found in the same object position, the higher its weight.
CHAPTER 3. MULTIMODAL OBJECT DETECTION

Figure 3.16: Weighting the shape flexibility for pedestrians. Codebook entries are weighted for their positional stability \( w^u_i \). Features found in the trunk are more stable (white), features found in the legs are less stable due to the part motion.

\( w^u_i \). We achieve this goal by just looking at the spatial vote distribution of each codebook entry. The weight \( w^u_i \) for the codebook entry \( i \), that casts \( K \) votes \( v_i \), is computed by equally considering the angle span and the vote distance:

\[
\begin{align*}
    w^u_i &= \frac{1}{2} \left( 1 - \frac{\arg\max(\angle(v_1, v_2), \ldots, \angle(v_{K-1}, v_K))}{\pi} \right) + \\
    &\frac{1}{2} \left( 1 - \frac{\arg\max(||v_1|| - ||v_2||, \ldots, ||v_{K-1}|| - ||v_K||)}{\arg\max(||v_1||, ||v_K||)} \right)
\end{align*}
\]

where the first term weights the stability of the angular position and the second term the distance stability of the entry. For example, in case of a pedestrian object, codebook entries found in the trunk receive high values due to its rigidness; features found in the limbs receive low values due to their flexibility and position change with respect to the object center (see Figure 3.16).

3.8.3.6 ISM Extension: High-dimensional Nearest Neighbor Search

Another problem of the ISM-based detector is the time required to produce codebook matching, that means the time spent in computing Euclidean distance between the codebook (and superfeatures codebook) entries and the image features. We provided a method to overcome this problem in Spinello et al. [2008a]. Image descriptors such as SIFT, SC or GLOH are very reliable (see Mikolajczyk and Schmid [2005] for a comparison), but they may have up to 128 dimensions. Performing a naive linear search of extracted descriptors
3.8. APPEARANCE INFORMATION FROM IMAGE DATA ANALYSIS

Figure 3.17: Subparts are computed by accumulating the angle of the interest points. The methods separates features in clusters when local minima in density is found, by using k-means with the BIC criterion.

(≈ 1000 in a 640 × 480 image) with a codebook of several thousand can be extremely slow, in the order of several seconds. Thus, a linear nearest-neighbor (NN) search cannot be used for real-time applications. KD-trees are the most effective solution to perform searches in data structures for small and moderate numbers of dimensions (2 to 20). As the dimensionality increases, they lose effectiveness, primarily because the ratio of the volume of a unit sphere in k-dimensions shrinks exponentially compared to a unit cube in k-dimensions. Thus exponentially many cells will have to be searched within a given radius of a query point, say for nearest-neighbor search. Also, the number of neighbors for any cell grows to 2k and eventually become unmanageable. In order to overcome this difficulties an exact Euclidean locality sensitive hashing is implemented. It provides a randomized solution for the high dimensional near neighbor problem in the Euclidean space \( L_2 \). Precisely, we used E2LSH, based on the work of Andoni and Indyk [2006], that is a method to perform a query in the codebook in sublinear time where each neighbor is reported with a certain probability. For the set of descriptors contained in the codebook \( \mathcal{I} \) and a given radius \( \tau_{\text{mathcall}} \), it finds all points \( d_i^c \in \mathcal{I} \) for a query point \( d_j \) so that
CHAPTER 3. MULTIMODAL OBJECT DETECTION

Algorithm 4 Estimation of the number of clusters for K-means clustering

Require: $b_i \in B$

1: $k \leftarrow 1$
2: while $\text{BIC}(B) \neq \max(\text{BIC}(B))$ do
3: Run clustering: $\text{kmeans}(k, B)$
4: $\text{FIT} \leftarrow \text{RSS}(\text{kmeans}(k, B))$ [RSS: Residual Sum of Squares]
5: $\text{BIC}(B) = -2 \ln (\text{FIT}) + k \ln (\|B\|)$
6: $k \leftarrow k + 1$
7: end while

$\|d^{C}_i - d^{I}_j\| \leq \tau_{\text{mathcall}}$ with a probability of at least $p$. In particular, E2LSH tends to have performance problems for data sets in which the high-dimensional space is not evenly populated and many approximate nearest neighbors could be found.

3.8.3.7 ISM Extension: Learning Explicit Subparts

The aim of this procedure is to enrich the voting process information by distinguishing between different object subparts from which the vote has been cast. This novelty has been introduced in (Spinello et al. [2009b]). This procedure is computed offline during the training process for each object category.

The idea is to define object subparts by segmenting the object in circular sectors. This segmentation choice is justified by the reason that an ISM object is defined as a learned set of descriptors and object center-voting vectors: an intuitive solution for defining subparts is then to group voters by their voting vector angle. We consider the accumulated interest point training set distribution $O^+$ (as we have done in Section 3.8.3.2) and we store the angle of each feature with respect to the horizontal line crossing the object center in the set named $B$, see Figure 3.17. The idea is to obtain groups of features that are separated when a local minima in density of the angular interest point distribution is found. Instead of using histogram discretization we found a solution by employing $k$-means clustering (Hartigan and Wong [1979]). $k$-means is a clustering method to partition $n$ observations into $k$ clusters in which each observation belongs to the cluster with the nearest mean. The biggest drawback of this method is that the number $k$ of clusters has to be set beforehand. We overcome this problem by using a probabilistic index called Bayesian Information Criterion (BIC), introduced by Schwarz [1978]. BIC is a criterion for model selection among a class of parametric models with different numbers of parameter, intuitively it balances the model error (by avoiding overfitting) and the model complexity. Thus, in order to determine $k$ we created an iterative estimation process described in Algorithm 4.

It is important to notice that the Residual Sum of Squares (RSS), for $k$-means, is a monotonically decreasing function with respect to $k$. RSS is exactly 0 when $k = \|B\|$, that is when each data point is a cluster by itself. BIC balances this effect with the model cost: the estimation process stops when the BIC value is maximized. It is important to notice
that the data lies in a space with a discontinuity: the interest point found at angle $2\pi$ must be treated as a point found at angle 0. We propose a solution to this problem: we consider the data located on a circle of fixed size and we employ, as distance for k-means, the arc length between two points. The resulting segmentation in clusters defines the parts subdivision in circular sectors for the object category. It can be formulated as a set of angle intervals $\mathcal{A} = \{(\alpha_s, \alpha_e)_1, \ldots, (\alpha_s, \alpha_e)_B\}$, where $(\alpha_s, \alpha_e)$ are the starting and the ending angle of a subpart and $B$ represents the number of clusters.

Note that this subpart extraction does not guarantee a semantical subdivision of the object (e.g.: legs, arms, etc. in case of the pedestrian object category) but it is interesting to see that it nevertheless resembles this automatically without interaction by the user (see Figure 3.18). In Section 3.8.3.9 we explain how to use this extended shape information for hypothesis selection.

### 3.8.3.8 ISM Extension: Learning Shape Templates

In Section 3.8.3.1 we explained that the training images are labeled by a binary image named segmentation mask. This mask has the size of the object’s bounding box and is 1 inside the shape of the object and 0 elsewhere. The idea of this extension is to create a probabilistic averaged shape of an object category by using the shape information contained in the training set segmentation masks. This potentially provides a tool to remove feature outliers, located in unlikely areas of the shape (see Section 3.8.3.9).

This procedure is technically simple and it is achieved in two steps. Firstly, all the segmentation masks for a given object class are overlayed so that their centers coincide. Thus, they are averaged, in order to obtain a template mask $T_c$ for each object class (see Figure 3.18, in white). This procedure is similar to the one used to prepare eigenfaces (Sirovich and Kirby [1987]), without subtracting the mean. Each $T_c$ is learned at scale 1 and each point contains a value between $[0, 1]$. If a scaled template is needed, $T_c$ is resized by using a bilinear interpolation.

Another intuitive method to obtain an averaged shape is to use the average of the pedestrians’ silhouettes or to produce a hierarchy of silhouettes. Our method does not rely just on the silhouette boundary but takes into account (probabilistically) the entire shape of the object category, thereby softening hard shape decisions by averaging. It is worth to mention that also Leibe et al. [2005] uses a shape verification method for refining object hypotheses, based on Chamfer matching (Borgefors [1988]) and silhouette shapes.

### 3.8.3.9 ISM Extension: Hypothesis selection

A ISM/ISMe detector produces objects hypotheses in several steps (see Section 3.8.3.3 and 3.8.3.4):

- a new image is retrieved
- interest points and relative local image descriptors are computed
- descriptors are matched with the codebook of a certain object category (e.g. pedestrian)
voted for objects centers are cast in a continuous 3D voting space defined by $x$, $y$, and scale coordinates.

- modes in the voting space are estimated by mean-shift with a balloon kernel estimator 

Thus, an object hypothesis is a locus in the voting space in which a high density of votes is present. Therefore, the task is then to select which hypothesis correctly represents an object.

**Evaluating vote distributions**

In the original definition of the ISM there is no assumption made on the particular shape of the objects to be detected. This has the big advantage that the learned objects are detected although they might be occluded by other objects in the scene. However, the drawback is that usually there is a large number of false positive detections in the image background. An hypothesis selection is needed. Leibe et al. [2005] addresses this problem by using a minimum description length (MDL) optimization based on pixel probability values. However, this approach is rather time demanding and not suited for real-time applications. Therefore, we suggested a different approach in (Spinello et al. [2008a]).

First we evaluate the quality of a hypothesis of a detected object center $\hat{x}$ with respect to two aspects: the overall strength of all votes and the way in which the voters are distributed. We can estimate the spatial distribution of voters $x_i^j$ that vote for $\hat{x}$ using a 1D circular histogram that ranges from 0 to $2\pi$. We compute the weight of the vote, according to equation (3.19), and also the angle $\alpha$:

$$
\alpha(x_i^j, \hat{x}) = \arctan2(y_i^j - \hat{y}, x_i^j - \hat{x})
$$

(3.33)
We store the voting weight in the bin that corresponds to $\alpha$. This way we obtain a histogram $\xi(\hat{x})$ with $B$ bins for each center hypothesis $\hat{x}$. The score $\sigma$ of a hypothesis is defined as the sum of all bins, i.e. the number of all voters for the object center. Now we can define an ordering on the hypotheses based on the histogram difference

$$\Delta_b(\hat{x}_i, \hat{x}_j) := \sum_{b=1}^{B} \xi_b(\hat{x}_i) - \xi_b(\hat{x}_j),$$

where $\xi_b(\hat{x}_i)$ and $\xi_b(\hat{x}_j)$ denote the contents of the bins with index $b$ from the histograms of $\hat{x}_i$ and $\hat{x}_j$ respectively. We say that hypothesis $\hat{x}_i$ is stronger than $\hat{x}_j$ if $\Delta_b(\hat{x}_i, \hat{x}_j) > 0$.

With this approach we are considering as more probable object hypotheses the ones that receive a big amount of votes from all around the shape.

**Improving the selection**

In Spinello et al. [2009b] we made use of all the information we collected with the methods explained in Section 3.8.3.2, 3.8.3.7 and 3.8.3.8 to further improve the hypothesis selection. We use, for each object category, the information of the learned subparts $A$ for defining the parameters of the histogram $\xi(\hat{x})$. Thus, we set $\|A\|$ as the number of bins of the histogram $\xi(\hat{x})$. We define the angle range of each bin by using the subpart angle intervals, described by the elements $[\alpha_s, \alpha_e)_i$ of $A$.

As a first step for hypothesis selection, we aim to refine the score of the hypotheses by using the learned shape templates. The votes that come from potential feature outliers might bias the hypothesis selection by assigning, by their presence, a high hypotheses score. We discard the votes that are implausible by using the information contained in the template mask: we place and rescale the shape templates at the object hypotheses position and then we assign a weight to each vote. Therefore, the voting equation (3.19) is weighted by the probability contained in the points of the template $T_c$:

$$p(o, \bar{X}_i | x^I_j, d^I_j) = p(o, \bar{X}_i | d^C, x^I_j)p(d^C | d^I_j)T_c(x^I_j, y^I_j)$$

(3.35)

Unlikely votes, outside the shape template, receive a very low weight and their bias contribution to the hypotheses score is greatly diminished.

Secondly, we insert all votes into the circular histogram $\xi(\hat{x})$, that has a bin for each sub-part of the object. To find the best hypothesis we define a partial order $\prec$ based on the function $\Delta_b$:

$$\hat{x}_i \prec \hat{x}_j \iff \Delta_b(\hat{x}_i, \hat{x}_j) < 0 \quad \text{where} \quad \Delta_b(\hat{x}_i, \hat{x}_j) := \sum_{b=1}^{B} \text{sign} (\xi_b(\hat{x}_i) - \xi_b(\hat{x}_j))$$

(3.36)

where $\xi_b(\hat{x}_i)$ indicates the value contained in the bin $b$ of the histogram for the hypothesis $\hat{x}_i$. Using this equation, we select the hypothesis with the highest order (in case of ambiguity we use the one with the highest strength). The hypothesis top ranked, after the partial sorting, is marked as a detection. Then, we remove all the votes coming from those features that contributed to the winning hypothesis, because we assume that an
image feature belongs to just a single object. Detection likelihoods are then updated until a minimum score threshold $\sigma_{\text{min}}$ is reached. A visual explanation of hypothesis selection method is shown in Figure 3.19.

In case of multiclass detection, we find the best hypothesis across all classes as described below in Section 3.8.3.10, remove all its voters and recompute the ordering.

Figure 3.19: Hypothesis selection using explicit subparts. Two hypotheses of category pedestrian are here evaluated. Votes are received for hypotheses generation (upper right). The votes are analyzed by using explicit subparts and templates (bottom left). Votes are accumulated in an histogram; votes coming from unlikely object parts (red arrows in bottom left) are less considered. The histogram has as many bins as the explicit subparts and accumulates the votes casted by voters at different angles. Finally the cost function is evaluated and hypotheses 2 is selected.

3.8.3.10 ISM Extension: Multiclass Hypothesis selection

In case of multiple object detection with categories, e.g. if we want to simultaneously detect cars and pedestrians, we need to formulate a technique to select which hypothesis from which class is the most likely. We run in parallel all the hypothesis selections, described by the partial ordering of equation (3.36), for each single object category, independently from each other. Then, we compare all the selected hypotheses from each category and decide which is the strongest.

Different objects categories are expressed by codebooks containing different amount of entries in the codebooks. Thus, the problem of comparing hypotheses from different
object categories arises due to the dependency of the scoring and histograms techniques used. We find a solution by comparing the relative areas covered by the voters from each class hypotheses. More precisely, we define a square area $\gamma$ around each voter that depends on the relative scale of each vote. The fraction of the area covered by all voters of a hypothesis and the total area of the object (computed from the template shape) is then used to quantify the inter-class hypothesis score. Care has to be taken in the case of overlapping class hypotheses. Here, we compute the set intersection of the interest points in the overlapping area and assign their corresponding $\gamma$ values alternately to one and the other hypothesis.

3.8.3.11 ISM Extension: Early hypotheses pruning in urban environments

A common problem of ISM based methods is the tendency of generating a high quantity of false positives. In Spinello et al. [2008b] we introduced an Early Stage hypotheses Pruning technique (ESHP).

In the voting stage an image feature can match several codebook entries and therefore it can vote for multiple object centers. Due to object symmetries, feature mismatches and scene configurations (i.e. vertical structures, buildings, posts and traffic signs) strong false positive object hypotheses may occur in empty or unlikely areas on the image. Therefore, we proposed an effective and fast way to remove this kind of errors based on a distance transform computation. The distance transform is an operator normally applied to binary images. The result of the transform is an image in which the value of each pixel contains the distance, in pixels, to the closest edge. In a distance transformed image, locally connected maxima form ridges and locally connected minima form valleys. The idea is that a large connected ridge represent an area that can be discarded because it does not contain any gradient information, which is a necessary condition for the detection of an object. Two additional parameters are required: the minimal area $I_q$ of a ridge that can be discarded, and a safety distance $I_w$ between a pixel and the edge that is closest to it in the image. Both of these parameters are set so that no contour of a pedestrian is included in the discarded area. This method is particularly effective in urban environments where roads and sky are often visible and contain no or little information. It consists in the following four steps (see Figure 3.20):

1. Compute a binary edge map by using Canny edge detector (Canny [1986]).

2. Compute an approximate distance transform from the binary image (Borgefors [1986]).

3. Select empty regions whose pixels have a distance of at least $I_w$ to the nearest edge. This is done efficiently in the distance transformed image by using connected component clustering on the ridges that have pixels with values more than $I_w$.

4. Discard all regions with an area that is bigger than $I_q$. The remaining image constitutes the region of interest for the hypotheses pruning.
Figure 3.20: Early hypotheses pruning by region of interest generation in urban environments. Uninformative content is discarded from the image by reasoning on the distance transformed image. **Top-Left:** Original Image. **Top-Right:** Edge image (Canny). **Bottom-Left:** Approximate distance transform. **Bottom-Right:** Result of the clustering in the distance transform image: hypotheses in the red area are discarded.

The only assumption we make is that a sufficient contrast is present in the image, which is reasonable, because object detection is generally hard in low contrast images. We make use of this region of interest in order to remove all the hypotheses that are located in empty areas, before the hypothesis selection is computed.
3.9 Sensor Fusion: combining information of Vision and Laser detectors

In this Section we describe the techniques used to combine the vision and the laser based detector. The contributions of this section have been presented in (Spinello and Siegwart [2008a], Spinello et al. [2008a]). In our work we employ an early fusion technique to discard object hypotheses and we use probabilistic sensor fusion methods to combine the outputs of the multimodal classifiers.

3.9.1 Camera and Laser Extrinsic Calibration

A very important factor in our multisensor system is the extrinsic calibration between camera and laser. We have employed the method explained in Pless and Zhang [2004] to calibrate a linear laser rangesensor with a camera. This method requires the system to observe a checkerboard in several poses by simultaneously capturing data with camera and laser. The checkerboard image and the relative laser scanline data give an information about the relative position and orientation of the camera with respect to the laser range finder. The transformation between camera and laser is obtained by using a quadratic error minimization criterion. It is important to remark that with the laser it is possible to define the horizontal limits of the checkerboard, therefore the vertical offset has to be set manually by knowing the placement height of the laser sensor. A laserscan line is projected in the image in Figure 3.21, after a successfull calibration.

3.9.2 Early fusion: using laser segments to define voting space boundaries

3.9.2.1 Early fusion with HOG

The early fusion step has been used to put constraint in the HOG object detection image search. Instead of running the algorithm at multiple scales all over the image, we defined a region of interest by projecting the laser segments as planes of fixed height in the image. This reduces the computation time and the amount of false object hypotheses.

3.9.2.2 Early fusion with ISMe

The early fusion step has been used to put constraint in the ISMe voting space (see Section 3.8.3.9), when generating object hypotheses. The idea is to project laser segments as 3D boxes in the voting space. If we consider a single laser segment, it is projected as a box with height set to a fixed value, width defined by the extremal points of the segment $S_i$ and depth is defined by the scale tolerance $\delta_{S_i}$. These 3D boxes define a boundary in the voting space for hypothesis selection for the vision detector. Before the hypothesis selection (see Section 3.8.3.9) is run, the early fusion takes place and it removes hypotheses that are not compatible with the boundaries.

In order to achieve this goal, we need to set $\delta_{S_i}$ for each object class. Precisely, we need to find $\delta_{S_i}$ as a function of the laser segment distance. The training image set is a
Figure 3.21: Accurate calibration is important for our multimodal system. An Alasca XT four laser layer scanline is projected in the camera image. Notice that a curb produces an offset in the projection, specially in far away points.

collection of height normalized images where an object category (e.g. side view of cars, pedestrians) has been labeled. In order to achieve a scale-distance calibration we need to compute a regression between the two variables. We assume, for practical reasons, that the relationship between two variables is linear, even though this is not true due to lens distortions. The non linearity in the regression is discarded specially because the vision based classifier assumes pedestrians of same normalized pixel height to be trained at scale 1, even though they have different height in reality. In order to fulfill a scale-distance calibration procedure, an object category is considered and a set of object’s camera images and corresponding laser segments are collected at several distances. Then, for each sample, each object pixel height $\omega_i^h$ is measured and the scale $\omega_i^s$ is calculated:

$$\omega^s = (\omega_1^s, \ldots, \omega_n^s)^T \quad \text{where} \quad \omega_i^s = \frac{\omega_i^h}{\omega_{\text{train}}^h} \quad (3.37)$$

where $\omega_{\text{train}}^h$ is the normalized height used in the training set of the object category. Similarly, for the laser data associated to the objects in the image, we store in $\omega^d$ the distance in meters from the origin to the center of mass of each laser segment. The idea is to produce a linear least squares regression that relates the objects pixel heights $\omega^s$ and
objects distances $\omega^d$:

$$\omega^d_i = \beta_1 \omega^s_i + \beta_2$$  

(3.38)

where $(\beta_1, \beta_2)$ are the unknown parameters of the line. Thus, we are able to query an hallucinated distance for each object category, by providing a scale input (and vice-versa). In order to produce such regression, we rewrite (3.38) in matrix form:

$$\beta = (\beta_1, \beta_2)^T$$  

(3.39)

$$\omega^d = \Omega^s \beta$$  

(3.40)

$$\Omega^s = \left( \omega^s \ 1 \right)$$  

(3.41)

where 1 consists in a column of ones. We use the pseudoinverse matrix to solve (3.40):

$$\beta = \left( (\Omega^s)^T \Omega^s \right)^{-1} (\Omega^s)^T \omega^d$$  

(3.42)

The scale $\omega^s$ estimated for each segment distance is then converted to the depth of the generated voting space 3D allowable region, in order to easily prune false hypotheses:

$$\vartheta_{S_i} = (\omega^s_i - \vartheta^*_{S_i}, \omega^s_i + \vartheta^*_{S_i})$$  

(3.43)

where $\vartheta^*_{S_i}$ is a constant fixed beforehand. Finally, an object center hypothesis is valid if it is inside of a allowable region. A visual explanation of the 3D allowance boxes can be seen in Figure 3.22.

The last step of the early fusion technique is to solve the data association problem of the assignment between segments and corresponding image hypotheses. We assume that each segment belongs to a single object. For each segment we compute the distance and compute an hallucinated scale (see equation 3.38). We solve the assignment problem in a greedy manner: given a segment $S_i$, the valid hypothesis, among all the hypotheses found in the projected segment volume, that minimizes the difference between the hypothesis scale and hallucinated scale is assigned to the segment $S_i$. The rest of the processing of hypothesis selection follows the technique explained in Section (3.8.3.9).

### 3.9.3 Early fusion: using 3D laser scan to improve space boundaries

A problem in urban environment, related to the generation of the allowable 3D regions in the voting space, is the placement of the ground plane. This factor can be crucial specially in areas where steep slopes and curbs are present. In such cases, the standard vertical placement of the allowable 3D box in the voting space is wrongly located; consequently, objects are wrongly discarded by the early fusion method (see Section 3.9.2). In Spinello et al. [2008b] we introduced a way of overcoming this problem by improving the placement of the regions of interest generated by the early fusion technique. We mounted a 3D laser scanner device (see Appendix section A.2) on the rooftop of the mobile platform and we gather a point cloud. The 3D rotating scanner device retrieves data from the full $360^\circ$ environment of the vehicle. The idea is to use this information to extract the position of
the ground plane in the local environment of the vehicle. We are then able to position, in the vertical axis of the voting space, the area of interest generated by the laser segments.

In the literature, there exist many different approaches to detect planes in 3D range data. The most important ones include region growing (see e.g. Hähnel et al. [2003a], Weingarten and Siegwart [2005]), Hough transform (see Iocchi et al. [2000], Okada et al. [2001]), split-and-merge (Kohlhepp et al. [2003]) and probabilistic estimation using expectation maximization (EM) (Liu et al. [2001], Triebel et al. [2005]). For the application described here, the major restriction is the required computation time, as we want to detect and track persons if possible in real-time. Therefore, we decided to use a simple
but time efficient region growing technique to detect the ground plane. The criterion for a scan point to belong to the ground plane is that its corresponding normal vector deviates only slightly (in our implementation by maximal 25°) from the upright vector $(0, 0, 1)^T$ and that it is not farther away from its closest neighbor than a given threshold (we use 1m). The region growing is initiated always at the same fixed point right in front of the vehicle at the ground level. To efficiently compute the normal vectors, we exploit the fact that the point clouds are structured in slices – each scan line of the vertically mounted rotating laser scanner accounts for one slice. This facilitates a fast and simple mesh triangulation performed by connecting two consecutive points from one slice with one point of the consecutive slice. From this triangulation the normal vectors are easily computed from the normalized cross product of difference vectors. An example result of the ground plane extraction is shown in Figure 3.23.

### 3.9.4 Probabilistic fusion: combining detectors

The scope of this work on multimodal object detection is to provide an information to a navigation or to a driving assistance module. For this reason, a natural output choice for our detector is to label laser segments with their class probability. Therefore, our fusion methods combines the detectors information and provide an output that consists in laser segments position and object category label. Two main approaches have been used for combining the output of vision and laser classification: firstly we developed a system based on Bayesian sensor fusion modeling (developed in (Spinello and Siegwart [2008a])), then we added tracking of laser segments with multiple motion models (developed in (Spinello et al. [2008a])).
Combining detectors by using Bayesian fusion

In the first work about pedestrian detection (Spinello and Siegwart [2008a]) we provided a way of combining the information coming from the HOG based detector (see Section 3.8.2) and the laser segments classifier (see Section 3.6). No time integration is here implemented.

We define the position of the centroid of a segment \( S_i \) in range data as \( \hat{x}^S_i \). Therefore, we define the prior of a pedestrian detection at a certain location as:

\[
p(o, \hat{x}^S_i) \approx p(o, \hat{x}^S_i | f_d)
\]  

Equation (3.44) modulates the prior uncertainty with respect to the distance \( f_d \) of the laser segment. This follows the intuition that the quantity of information retrieved by the sensors decreases with respect to the distance: a far away object is described by a small amount of points in the range data and a small amount pixels in the image data. Therefore, a far away object should receive a lower value prior with respect to a closer one:

\[
p(o, \hat{x}^S_i | f_d) := 1 - \frac{f_d(S_i)}{f_{d}^{max} + \eta_d}
\]  

where \( f_{d}^{max} \) is the maximum sensor distance, \( \eta_d \) is a small constant that avoids a zero prior at max range.

We define \( p(o, \hat{x}^S_i | F) \) as the detection probability, based on laser data features \( F \), of the segment \( S_i \). This value, namely the structure information probability, is computed by using equation (3.6). Thus, we define the appearance information probability as the probability of pedestrian detection given the image data features \( F_{HOG} \):

\[
p(o, \hat{x}^S_i \rightarrow \hat{x}^I_j | F_{HOG})
\]

where \( \hat{x}^S_i \rightarrow \hat{x}^I_j \) describes the association between laser segment \( S_i \) and object hypothesis \( \hat{x}^I_j \). The mechanisms of the association between laser segments and camera hypotheses have been presented in Section 3.9.2.

The information fusion is addressed by employing a simple Bayesian modeling approach. The usage of the Bayes formula, when all classifiers are considered to be independent, yields to the following decomposition of the conditional probability:

\[
p(o, \hat{x}^S_i | F, F_{HOG}) = \frac{1}{Z} p(o, \hat{x}^S_i | F) p(o, \hat{x}^S_i \rightarrow \hat{x}^I_j | F_{HOG})
\]

where \( Z \) is the probability normalizer. The outcome of equation (3.47) is assigned for each laser segment, and it represents the sensor fusion. It is interesting to notice the similarities between equation (3.47) and the probabilistic fusion formula shown in (2.8). They both produce information fusion by using the Bayes formula. The concept of fusing appearance and geometrical information is present in both equations; the difference between the two formulas consists just in the way the prior and the likelihoods are computed.
3.9. SENSOR FUSION

3.9.4.2 Combining detectors by using Multiple Motion Models Kalman Filters

In the work of (Spinello et al. [2008a]), we introduced tracking as a mean of integrating class probabilities over time and as an additional algorithm output, to provide prediction information. Our needs are to design a reliable tracking method that does not rely on a single tracking hypothesis but also that scales gracefully with the number of objects to track. Several methods exists in the literature (MHT and JPDA based methods to cite the most famous, see Cox [1993] for a summary) but we would design a tracker that keeps simplicity and copes well with the motion model of several kinds of objects categories. Pedestrians are hardly described by motion models: they can stop, suddenly turn on spot, inverse their trajectory etc; cars have instead a precise motion model (Ackerman steering (Ackermann [1818])). Therefore, we introduced a tracker design in which each track is described by multiple Kalman Filters, each providing a different motion model. The advantage of this method is that the number of estimating filters scales linearly with respect to the number of object to track. Moreover multiple hypotheses regarding objects motions are also produced for each time step.

Tracks are managed by a tracking manager that solves data association, creates or deletes tracks. We assume that each track is associated maximum to a single segment, that \( N \) is the number of laser segments \( S_i \), \( R \) is the number of tracks and \( M \) is the number of Kalman Filters present for each track, each with a different motion model. Data association, the assignment problem of assigning laser segments to tracks, is solved in two steps. The first step is to compute which motion model to use for each track. In each track, the distance between the Kalman Filters (KF) prediction and the \( N \) laser segments centroids are computed. This process generates for each track, \( M \) Mahalanobis distances for each observation (Mahalanobis [1936]). In each track, the closest distance, for each observation is taken and the KF generating such prediction tagged. At the end of the first step of laser segments data association, every track obtains a set of \( N \) distances from \( N \) observations.

The second step of data association is used to select which observation is assigned to which track. A combinatorial optimization method called Munkres assignment problem is formulated to solve this problem. We want to assign \( N \) hypotheses to \( R \) tracks (where \( N \neq R \)). A rectangular matrix of size \( R \times N \) is generated in which rows represent tracks indices and column observations indices. The previously computed distances are inserted as values of the assignment matrix. The solution of the combinatorial minimal weight assignment has been found with the extension of the Munkres method for rectangular assignment matrices proposed by Bourgeois and Lassalle [1971]. If there are more segments than tracks then \( N - R \) new tracks are created. Instead, if more tracks then segments are present in a certain moment, the tracks that are not updated with a new observation are maintained until their variance in \((x, y)\) reaches a certain maximum threshold \( \delta_{x,y} \).

We now give a mathematical formulation for the tracks and for the fusion of the detection outputs. We track cluster centroids in the 2D laser coordinate system by using two system states, one for each motion model:

\[
\begin{align*}
\mathbf{x}_{m1} &= \left( \hat{x}^S, \hat{y}^S, \dot{\hat{x}}^S, \dot{\hat{y}}^S, (p_1, \ldots, p_C) \right) \\
\mathbf{x}_{m2} &= \left( \hat{x}^S, \hat{y}^S, \dot{\hat{x}}^S, \dot{\hat{y}}^S, (p_1, \ldots, p_C) \right)
\end{align*}
\]
Here, \((\hat{x}^S, \hat{y}^S)\) is the velocity of the cluster centroid \((\hat{x}^S, \hat{y}^S)\) and \(p_1, \ldots, p_n\) are the probabilities of all C classes. The observation vector \(z(k)_i\), at time \(k\), consists of the position of the cluster and the class probabilities for each sensor modality:

\[
z_i = (\hat{x}^S_i, \hat{y}^S_i, (c_1, \ldots, c_n)^T, \ldots, (p_1, \ldots, p_C)^T)
\]  

(3.50)

Here, \((\hat{x}^S, \hat{y}^S)\) is an observation of a cluster center and \(z\) denotes the number of sensors. Each block \((p_1, \ldots, p_C)\) is the estimate given by the laser or camera based classifier. This kind of estimation procedure resembles tracking-before-detection techniques in which segments are tracked and the correct object category is estimated through time.

Henceforth, we can write the matrix needed for the Kalman Filters. For the prediction step at time \(k\):

\[
x(k)_{mi}^- = A_{mi} x(k - 1)_{mi}^- 
\]  

(3.51)

We can write the state matrix \(A_{mi}\) in the case of two motion models and two classes:

\[
A_{m1} = \begin{pmatrix}
1 & 0 & \Delta k & 0 & 0 & 0 \\
0 & 1 & 0 & \Delta k & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

(3.52)

\[
A_{m2} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0
\end{pmatrix}
\]

It the matrix \(V^1\) indicates the state covariance matrix and \(V^2\) the sensors covariance matrix, we compute:

\[
P(k)_{mi}^- = A_{mi} P(k - 1) A_{mi}^T + V^1
\]  

(3.53)

The update step is calculated by computing the Kalman gain \(G\), then updating each state \(x(k)_{mi}\) and the covariance matrix \(P\):

\[
K_{mi} = P(k)_{mi} G^T \left( G P(k)_{mi} G^T + V^2 \right)^{-1}
\]

(3.54)

\[
P(k)_{mi} = (I - K_{mi} G) P(k)_{mi}^{-}
\]

(3.55)

\[
x(k)_{mi} = x(k)_{mi}^- + K_{mi} (z(k)_n - G x(k)_{mi}^-)
\]

(3.56)

where \(z(k)_n\) represents the assigned observation vector to the track. The matrix \(G\) models the mapping from states to the predicted observation and is defined as \(G = (G^T_s G^T_{s1} \ldots G^T_{sc})^T\), where \(G_s\) maps to pose observations and the \(G_{si}\) map to class probabilities per sensor. For example, for one laser, one camera and constant velocity we have:

\[
G_s = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

\[
G_{s1} = G_{s2} = \begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

(3.57)
3.10 Experimental evaluations of visual-based detection and sensor fusion

In this section we provide experimental evaluations of the methods regarding image-based object classification, sensor fusion and tracking techniques. We evaluate both image-based object classification and fusion techniques because visual detection depends also on fusion techniques. In particular, it depends on early fusion techniques, for example, to restrict the image search space (see Section 3.9.2).

Data logs have been taken with our urban mobile robotic platform. Experiments have been conducted by using the sensor setup configurations presented in the Appendix section A.3.

3.10.1 Real World Datasets: Image data

We have presented in Section 3.7.2 several datasets for evaluating the different proposed techniques in range data. Laser data and image capture is synchronized, therefore each of the presented dataset (UD1, UD2) contains as many images as laser scans. The imagery is manually labeled with rectangle boxes indicating pedestrians and cars. Annotations are marked if at least half of an object is shown or the object width is bigger than 80 pixels. A suite of MATLAB scripts have been used to simplify this process.
3.10.2 Evaluating SBA HOG classification and Bayesian fusion for Pedestrian Detection

We here evaluate the quality of object detection in range data by using the SBA detection technique based on HOG features, explained in Section 3.8.2. The vision-based detector is combined with a SBA range-based detector (see Section 3.6.4, evaluated in Section 3.7.3) by using Bayesian modeling (see Section 3.9.4.1).

**Training**

We trained our HOG features classifier using the well-established MIT pedestrian image database (Mohan et al. [2001], Papageorgiou and Poggio [2000]) and the significantly more challenging INRIA person database (INRIA 06). A software has been developed to build the negative set by randomly cropping part of images containing streets and urban background from the INRIA negative dataset. The set contains in total 3123 $64 \times 128$ positive images and 12313 $64 \times 128$ negative images of people. The people are usually standing but appear in any orientation, against a wide variety of background including crowds.

The cascade consists in 26 levels, the first three levels contain just 4 to 6 SVM classifiers each and reject circa 81% of the detection windows. The low number of early classifiers allows a fast execution time and good performance (this is a comparable result with respect to the original implementation of Zhu et al. [2006]).

**Detection**

In this section we evaluate the performances of the SBA HOG classification technique. The detector is run on the image space restricted by projected laser segments, see early fusion techniques of Section 3.9.2.1.

In Table 3.6-left we show the confusion matrix of the HOG cascade classifier for the
parking lot testing set of UD1 dataset. The detection rate is very high (over 90%) and false positive rate/false negative rate is low (around 8% for both). Even though the environment resembles a road scenario, the visual background is simple and low textured and just a small amount of clutter data is present. This ease the task of the classifier that has to evaluate data with low classification ambiguity.

The second part of UD1 is based on the cluttered environment of a university campus. Results are given in Table 3.6-right.

Table 3.6: Evaluation of vision based classifier based on SBA HOG classification. In the table, GT is the acronym of Ground Truth. **Left**: Confusion matrix for parking lot dataset **Right**: Confusion matrix for university campus dataset.

<table>
<thead>
<tr>
<th>GT</th>
<th>Prediction</th>
<th>Tot</th>
<th>Prediction</th>
<th>Tot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>N</td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>119 (72.6%)</td>
<td>45 (27.4%)</td>
<td>164</td>
<td>197 (91.53%)</td>
</tr>
<tr>
<td>N</td>
<td>173 (21.9%)</td>
<td>613 (78.1%)</td>
<td>786</td>
<td>137 (7.62%)</td>
</tr>
</tbody>
</table>

The detection rate is around 73% for the HOG classifier. This lower performance result can be explained by the complexity of the environment and, moreover, by the clutter present in this scenario: multiple overlapping pedestrians may create severe occlusions that can affect the values contained in the HOG features. In general overlapping pedestrians contain less informative gradient information for the HOG detector. Moreover, a visually complex scenario is present, specially vertical visual structures created by buildings that create ambiguities for the HOG detector.

**Evaluating fusion** Instead of having an hard method of combining the results of both classifiers we presented a method based on the Bayesian sensor fusion approach (see Section 3.9.4.1). In Figures 3.24, 3.25, 3.26 each detection (in the range data and in the image) is labeled with its fused probability, far away clusters have a lower probability due the modulation factor of the prior present in equation (2.8). The probability output for each segment might help to create a reasonable dynamic traversability map for an autonomous car, taking into account the confidence of each sensor.

If one sensor detection fails to detect a person (false negative) the result is a decrease in the fused probability value in equation (2.8), but the affected cluster still receives another probability measure from the other sensor detector. This condition occurs mainly when a pedestrian is defined with few range data points or with an occluded or unconventional silhouette pose. In order to show the validity of the method we depict in Figure 3.27 the evolution of the probability estimate in time when a pedestrian is walking in front of the car. The brightness of the circle is proportional to the detection level. It is important to notice that the detection functions also when the person is not present in the image but it is estimated only using range data: when a laser segment is projected outside the image
boundary, a small fixed value is assigned to $p(o, \hat{x}_i^S \rightarrow \hat{x}_{i|F_{HOG}})$ and the Bayesian fusion is run.

Moreover, we evaluated the fused detection rate (true positive rate) by using image/laser synchronized sequences from the two sets in which pedestrians are present. The resulting detection rate increase for the first set is 2% and 16% for the second. It is important to notice that the smaller performance gain resulting from the first set is related to the already good quality vision/laser classifiers.

The computation time required to obtain a detection depends on the number of clusters found in the laser range data scan, because they define the processing image area for the vision classifier (see Section 3.9.2.1). A frame rate between 3 fps to 15 fps is obtained in the two datasets.

### 3.10.3 Evaluating ISMe classification for Pedestrian Detection and Tracking

We here evaluate the quality of pedestrian detection in image data by using the ISMe method, introduced in Section 3.8.3. It is important to remark that ISMe is run only in interesting image regions, defined by projected laser segments, see Section 3.9.2.2. Vision-based detector is combined with a SBA range-based detector (see Section 3.6.4, evaluated in Section 3.7.3) by using multiple motion tracking, see Section 3.9.4.2.
3.10. VISUAL-BASED DETECTION AND SENSOR FUSION EVALUATION

Figure 3.27: The figure shows the probability evolution of a pedestrian walking in front of the car. The left figure traces the path: brighter green circles depict a high probability value. On the right figure the probability value (axis Y) of the left figure is explicitly plotted with respect to the frames (axis X).

Training We trained our image detection algorithm ISMe by using a set of 400 images of persons with a height of 200 pixels at different positions, dressed with different clothing and accessories such as backpacks and hand bags in a typical urban environment. SIFT descriptors (Lowe [2003]) computed at Hessian-Laplace (Mikolajczyk and Schmid [2005]) interest points are collected for the codebook building. The codebook generation for the pedestrian detector is achieved by agglomerative clustering, in SIFT descriptor space, with an euclidean distance of 250. The resulting codebook is constituted by 8451 entries in 128 dimensions.

Detection In order to show a quantitative performance evaluation, several comparisons have been performed.

For the vision based classifier, a comparison on the dataset UD2.1 between ISM, the proposed extended ISM (ISMe) and an Haar based AdaBoost (HAda) detector is shown in the Precision-Recall graph of Figure 3.28. Equal error rates (EER) are highlighted in each curve in order to show the performance gain. It is important to notice that at high Precision values ISM (and HAda) shows a low Recall (less true positives), while our method, thanks also to the ESHP and the proposed extensions, performs much better. HAda in general shows the limit of using not robust Haar features for obtaining detection in complex backgrounds. An Haar feature considers the difference in pixels intensities of rectangular subdivisions in a rectangular region. Even though this feature is still widely used in object detection tasks, it is not suitable for visually complex environments. In our experiments HAda produces a lot of false negatives also due to the cascade organization of the detector: if just one stage fails then the result is a negative output. The ISMe curve is significantly flatter than the other two methods and tends to the optimal upper right
corner of the graph. To quantify: ISMe, ISM and HAda obtained respectively Precision 80%; 81%; 78% at Recall 63%; 22%; 0.01%. We also note that many annotated pedestrians are severely occluded, and the detection task is so difficult that a performance of over 90% is far beyond the state of current computer vision systems. Another important difference between our approach and the original ISM is the quantity of features to evaluate for each image. ISMe and ISM use just one kind of image feature (SIFT) and one type of interest point (Hessian-Laplace). Even though ISMe performs better than ISM, the latter is generally used with two or three kinds of descriptors (see Leibe et al. [2005, 2007]) to enhance robustness. Therefore, in average the number of descriptors to be matched and processed by ISMe is less than half than ISM.

Moreover, we plotted Recall-per-frame values over frames in Figure 3.29, in order to show a comparison between ISMe and HAda. We can notice, as we expected, that the AdaBoost based approach yields a very low hit rate; on the contrary, ISMe obtains a quite high true-positive rate during the entire frame sequence. Another performance evaluation is shown in Figure 3.30, where the quantity of false positives per frames are compared between ISMe and ISM. Here the difference is evident and it is interesting to see that the two graphs depict a clear advantage of using ISMe. This advantage comes specially from the ESHP and the early fusion process for hypotheses selection.

Experiments have been performed to evaluate the processing time of matching image features with an ISMe codebook by activating our fast high-dimensional hashing. We compared E2LSH (see Section 3.8.3.6) with a linear search. 1260 descriptors extracted

**Figure 3.28:** Comparative Precision Recall graph between ISMe, ISM and Haar based AdaBoost for dataset UD2.1. Equal error rate (EER) is respectively 69%; 61%; 26%. Notice the flatness of ISMe specially at high Recall values.
3.10. VISUAL-BASED DETECTION AND SENSOR FUSION EVALUATION

Figure 3.29: Image detection recall values for each frame in sequence 1 (left) and sequence 2 (right) for dataset UD2.1: ISMe vs Haar AdaBoost cascade detector. ISMe obtains a higher detection rate than the other method mainly due to the distinctiveness of the features used, the detection given by a soft decision on multiple votes and the robustness against occlusion.

Figure 3.30: False positives in image classification evaluated for each frame of sequence 1 of dataset UD2.1 compared to standard ISM at EER value. Here it is clear the advantage of using ISMe over ISM.

from random images of the two datasets have been matched with two pedestrian ISMe codebooks. One codebook is built with Shape Context descriptors (SC, 36 dimensions) and the other with SIFT descriptors (SF, 128 dimensions), both of the codebooks have been limited to contain maximum 1800 entries. We show, in Figure 3.31, for each box-plot, the
time elapsed for matching a feature descriptor with E2LSH and with a linear search, in case of SIFT and SC codebooks. Box-plots can be useful to display differences between populations without making any assumptions of the underlying statistical distribution. The red lines in the plot depict the medians, data points beyond the whiskers are marked as outliers, the box bottoms depict lower quartile and, box tops, upper quartile. It is noticeable that the simple linear search (SC\_NN and SF\_NN in Figure 3.31) obtains worse median times with respect to E2LSH search (SC\_E2LSH and SF\_E2LSH in Figure 3.31). It is also remarkable that the median search time increases slowly for E2LSH between SC and SIFT descriptors, even though it exists a big difference in terms of numbers of dimensions. The plot depicts also that just a small difference in terms of median matching time exists when using the lower dimensional Shape Context descriptors. The big variance of E2LSH matching in case of the SIFT codebook is explainable as the memory overhead and operating systems calls (kernel scheduling) that this method introduces. Even though Andoni and Indyk [2006] presented a more encouraging search time improvement (E2LSH showed to be between 1 and 29 time faster than linear approximate NN search) with several datasets, we obtained 0.5 to 2 times improvement. This might be caused by a known pitfall of E2LSH: hashing is dependent on data, if the matching radius of a query point contains many neighboring points then the algorithm results less performant.

Another experiment has been conducted to evaluate the improvement brought by early hypotheses pruning ESHP (see Section 3.8.3.11). 100 random images containing labeled pedestrians are selected from the test image datasets and a measure of hypotheses pruning quality is estimated with respect to the minimum area to discard $l_q$ and the image area $I_{area}$: we count the number of correctly pruned hypotheses (removed false positives $f_{p\text{prune}}$) divided by all the false positives $f_{p\text{tot}}$ and the number of remaining valid hypotheses (true positives $t_{p\text{good}}$) divided by all the number of positive hypotheses $t_{p\text{tot}}$. For convenience we plot these variables with respect to $l_q/I_{area}$. Figure 3.32 shows that the valid pedestrian hypotheses are not removed if the minimum area $l_q/I_{area}$ is big but, intuitively, at that value a low rate of false positives are pruned. The ESHP minimum area to discard has been set to $l_q/I_{area} = 0.075$, that is the point in which 99% of the true positives are maintained but 25% of false positives are removed. In a 640 x 480 image this represents an area of 23040 pixels, roughly a blob of 200 x 115 pixels. An ESHP segmented region, that means a region with no or very low gradient content, bigger than that size is forced to not contain pedestrian hypotheses. Intuitively, this area limit seems to be too small: pedestrians have to be detected also when they are close to the camera and they appear bigger than 200 x 115. Statistically, it rarely happens that pedestrians contains such an extended area with absolutely no (or extremely low) gradient information. Urban landscapes are visually complex scenes where visual clutter and textured objects are all around, so the practical effect of ESHP is often to remove the sky or the road surface. In other kinds of environments like in highways scenes, or in case we aim to detect other categories of objects (e.g. cars, trains), physically wider than pedestrians, ESHP might not be a suitable technique or it should be used with other additional constraints.
3.10. VISUAL-BASED DETECTION AND SENSOR FUSION EVALUATION

Figure 3.31: Codebook entry matching time comparison E2LSH vs linear search. Each box-plot in the graph depicts the time elapsed to match a descriptor with the ISMe codebook. Shape Context (SC) and SIFT (SF) based codebooks, composed each of 1800 entries, are evaluated with 1260 random feature descriptors extracted from the image set.

Evaluating fusion  The tracker is configured with two motion models for each Kalman filter: constant velocity and brownian motion. In order to show the usefulness of using a multimodal detection algorithm, we performed an experiment consisting of evaluating an object track: a person emerges from outside the sensors common field of view, walks in front of the sensors and then disappear. A fused probability is compared with respect to the output of the laser classifier and the image classifier, as shown in Figure 3.33. It is important to remark that the tracker assigns a higher confidence to the vision based detector, therefore the fused detection probability is biased towards the blue line. It is noticeable that the dynamic produced by the Kalman filter fusion, obtained by tuning the elements of the matrix $V^1$ relative to the class states (see Section 3.9.4.2), permits that small time sequences in which both detectors output have low confidence ($< 0.5$, at 5 – 6s in Figure 3.33) to mark the track as valid (class confidence $> 50\%$).

Qualitative results from two frames are shown in Figure 3.34. The box colors in the image describe different tracks, the size of the filled circle is proportional to the pedestrian detection confidence.
Figure 3.32: Evaluation of early pruning technique based on distance transform clustering. Two variables are evaluated: the amount in percentage of correctly pruned false positive hypotheses (in blue) and the percentage of valid hypotheses (in red). The x-axis represents, in percentage, the ESHP minimum area to discard. $l_q$ represents minimum image area to discard, $l_{area}$ the entire image area. Notice that the clustering is effective to reduce the amount false positives, without decreasing the number valid hypotheses, when $\frac{l_q}{l_{area}} \geq 0.075$.

3.10.4 Evaluating ISMe classification for Multiclass Detection and Tracking

We here evaluate the quality of multiclass detection in image data by using the multiclass ISMe method, introduced in Section 3.8.3. The two trained object categories are pedestrians and cars. It is important to remark that ISMe is run only in interesting image regions, defined by projected laser segments, see early fusion techniques Section 3.9.2.2. Vision-based detector is combined with a CRF, explained in Section 3.6.4, by using multiple motion tracking, see Section 3.9.4.2. For this work, ESHP technique has been disabled because of its sensitivity to objects with wide empty areas, cars in our case. In this section we perform comparisons of our multiclass technique with other detection methods for laser and vision based classifier.

Training Several ISMe codebooks need to be trained due to the complexity of the multiclass (cars, pedestrians) classification task. Experience shows (Leibe et al. [2007]) that
Figure 3.33: A single pedestrian is tracked. Comparison of the fusion tracker probability output (in green) with detectors output (in blue and red). Notice that small sequences of low detection likelihood from both sensors do not influence the validity of the track (fused probability > 0.5).

lateral views of pedestrians generalize well to front/back views. Therefore, we used the same set used in Section 3.10.3 as training data for the pedestrian detector. The class ‘car’ has been learned from 7 different viewpoints as in Leibe et al. [2007] (see also Figure 3.35, left). Car codebooks are learned using Shape Context (SC) descriptors (Belongie et al. [2002]) at Hessian-Laplace interest points (Mikolajczyk and Schmid [2005]). The pedestrian codebook uses lateral views and SC descriptors at Hessian-Laplace and Harris-Laplace interest points for more robustness. We selected SC instead of SIFT descriptors due to their lower dimensionality: this shortens the time for feature extraction, for the agglomerative clustering of the codebook generation and for feature matching with codebooks. In the work of Leibe et al. [2006], the author compares several descriptors for object detection and he shows that SC descriptors are very good features for object detection.

Detection First of all we run of our new multiclass detector on the pedestrian dataset UD2.1, in order to quantify the performance enhancement with respect to our previous work on pedestrian detection, presented in Section 3.10.3. Then, we run the experiments
**Figure 3.34:** Qualitative results from sequence 1 and 2 of dataset UD2.1 showing pedestrians crossing. The colored boxes in the image describe different tracks and probability levels; the size of the filled circle in the tracking figure is proportional to pedestrian detection confidence. It is important to notice that highly occluded pedestrians are also successfully detected and tracked.

on the larger set UD2.2.

The comparative results of our image detection algorithm with our previous work, on dataset UD2.1, are shown in Figure 3.37. Our vision based multiclass detection, named ISMe2.0 in the plots, is compared to the standard ISM, our previous single class detector ISMe1.0 and with an AdaBoost trained Haar features detector (ABH) for the class pedestrian. We can see that our method yields the best results. It is important to see that the multiclass method obtains higher recall values that the previous ISMe1.0, mostly due to the refinements introduced in the hypotheses selection step, namely the object subparts and shape templates (see Section 3.8.3.9).

We then run the system for the challenging dataset UD2.2. Pedestrian detection with camera is shown in Figure 3.36-left. We compared our image detector with respect to an Haar AdaBoost (ABH) based classifier and, in case of the pedestrian detector, with the Histogram of Oriented Gradients (HOG) technique that we employed in the first work (explained in Section 3.10.2). In case of HOG and ABH we used the early fusion
3.10. VISUAL-BASED DETECTION AND SENSOR FUSION EVALUATION

Figure 3.35: Left: For car classification, we use codebooks from 7 different views. For training, mirrored images are included for each view to obtain a wider coverage. Right: For pedestrians we use a codebook of side views with mirroring. Lateral views have sufficient information to generalize frontal/back views.

The technique explained in Section 3.9.2 in order to reduce the image search space. Our multiclass detector, shortly named ISMe, clearly outperforms the other methods, EER is: 60.01% for ISMe, 52.21% for HOG, 11.17% for ABH. In general, if one is willing to accept a high rate of false positives, the ISMe2.0 detector obtains over 70% Recall. At that values the difference with respect to the other methods is even more evident.

We then evaluated the performance of our system in case of car detection on the dataset UD2.2. The ISMe car image detector outperforms the ABH detector, the latter has been trained on trunks, sides and frontal view of cars. It is important to remark that the results shown in Figure 3.36-right are averaged between the 7 car views of ISMe. Equal Error Rate is crossed at 72.54% for ISMe and 18.93% for ABH. It is interesting to notice that cars are in general detected easier than pedestrians. The reasons are multifold: cars are rigid objects that have a fixed visual appearance, they often have a simple visual structure even though they present symmetries that could create voting ambiguities. Moreover, the dataset has been taken in urban environment, therefore the quantity of trunk/frontal views represent the majority of the car instances. Such views are the simplest among the other car codebooks.

Evaluating Fusion  The tracker is configured with two motion models for each Kalman filter: constant velocity and brownian motion. Tracking and fusion for the pedestrian category is evaluated in Figure 3.38. We show the precision-recall graph and a ‘Recall-false positives per frame’ plot in order to show the performance increase. It is interesting to see in the plot of Figure 3.38-left that the camera and laser detectors are very complementary sources of information: their combined contribution allows to produce a fused detection rate that is higher that each single one; this phenomenon is even more evident when
Our approach outperforms the other methods for both object categories. **Left:** The image based detection is compared with Histogram of Oriented Gradients detector (HOG) and an AdaBoost classifier with Haar features (ABH). **Right:** The image based detection is compared with AdaBoost classifier with Haar features (ABH).

Precision is low. Tracked and fused EER for pedestrian detection is 69.8%. In Figure 3.38-right we show such improvement in another form. If we fix a certain false positive rate per frame, we obtain a higher Recall value with the fusion method. Tracking and fusion for cars is shown in Figure 3.39. Similar conclusions to pedestrian fusion could be given.
3.10. VISUAL-BASED DETECTION AND SENSOR FUSION EVALUATION

Figure 3.38: Quantitative evaluation of tracking and fusion for pedestrian detection, dataset UD2.2. Precision-Recall graph (left) and ‘Recall-false positives per frame’ show that the fusion method enhances the results of single classifiers. For each frame \( \sim 2 – 3 \) objects are present.

Tracking allows a better detection rate than each single classifier and a reduced number of false positives per frame; EER is crossed at 78.4%. Fused EER is significantly higher than the results obtained for the pedestrian category due to the higher performances of vision and laser detectors.

Figure 3.39: Quantitative evaluation of fusion for car detection, dataset UD2.2. Precision-Recall graph (left) and ‘Recall-false positives per frame’ show that the fusion method enhances the results of single classifiers. For each frame \( \sim 2 – 3 \) objects are present.

Some qualitative results are shown in Figure 3.40 where a passing car and a crossing pedestrian are correctly detected and tracked. It is important to notice that even though images and laser data show very low contrast, partial occlusions and clutter, the system
manages to detect and track the objects in the scene.

Figure 3.40: Cars and pedestrian detected and tracked under occlusion, clutter and partial views in dataset UD2.2. In the camera images, left column, blue boxes indicate car detections, orange boxes pedestrian detections. The colored circle on the upper left corner of each box is the track identifier. Tracks are shown in color in the right column and plotted with respect to the robot reference frame.
3.11 Conclusions

In this chapter we have shown our contribution to multimodal objection detection method. This work represents the only approach in mobile robotics, to the author’s best knowledge, tackling the challenging task of detection and tracking based on laser and camera that does not rely on heuristics or manually set parameters.

We successfully detect and track several kinds of object categories (cars and pedestrians). Our method is largely based on machine learning approaches, specifically based on supervised learning. In our first work (Spinello and Siegwart [2008a]) we detected pedestrians by combining, with Bayesian fusion modeling, HOG image detectors and boosted cascades of features computed on Delaunay-connected segments. We then expanded this multimodal approach by using an Implicit Shape Model (SM) image detection approach. We contributed and evaluated several extensions to the original ISM in the field of feature selection and matching, subpart reasoning and selective weighting (Spinello et al. [2008a,b]). Moreover, we added a tracking method based on multiple motion Kalman filters in order to integrate the detector information in time. Finally, we extended this approach to classification and tracking of several object categories by combining Boosted Conditional Random Fields and a multiclass based ISMe (Spinello et al. [2009b]). We evaluated the performance, qualitatively and quantitatively, of our method by using real urban datasets in which clutter and multiple objects move in the field of view of our mobile vehicle.
In this chapter we present applications of scene analysis by using reasoning from single images. We present two works, the first is about pedestrian detection at very small scales (Spinello et al. [2009a]), the second is about finding repetitive patterns and exploiting this information for image compression or for inferring missing elements. These very different topics have in common that a single image is the only source of information for the algorithms.

Vision is a sensor modality that has enormous potentials. Sometimes an image is everything a robot can obtain to infer the solution of a certain problem. Far away pedestrians are hard to detect from a single image: just few pixels define the shape and textures are washed out by the limited image resolution. Moreover, the previously explained multimodal method (see Chapter 3) fails to obtain good results because laser sensors do not have enough angular resolution and image detectors are not reliable at very small scales due to interest points scarcity and limited image resolution.

We introduce in this chapter also a generic method to analyze repetitive patterns: a natural choice to show the potentiality of the technique is to apply it to repetitive patterns found in images. A little attention, until now, has been given to high level interpretation of patterns, like repetition of windows in a building facade or to the visual sequence of columns on a temple. Therefore, we then show how to use this technique to infer missing elements and to provide a high level compression of an image.

4.1 Detecting Pedestrians at Very Small Scales

This work, presented in Spinello et al. [2009a], presents a novel image based detection method for pedestrians at very small scales (between 16 x 20 and 32 x 40).
We propose a set of new distinctive image features based on collections of local image gradients grouped by a superpixel segmentation. Features are collected and classified using AdaBoost. The positive classified features then vote for potential hypotheses that are collected using a mean shift mode estimation approach. The presented method overcomes the common limitations of a sliding window approach as well as those of standard voting approaches based on interest points. Extensive tests have been produced on a dataset with more than 20000 images showing the potential of this approach.

We already introduced in Chapter 3 the risk for pedestrians in busy cities and in traffic situations. In (tcs [2007]) two major trends can be observed: first the steady decrease in the total number of dead and seriously injured persons over the last 30 years, and second the increase in the percentage of dead and injured pedestrians. The former is mostly due to the growing number of safety systems available for modern vehicles, while the latter originates from the fact that primarily motorists and cyclists benefit from such safety systems, but not pedestrians. One way to address this problem is to build more intelligent driver assistant systems that aim at protecting the driver and the pedestrian and avoid a potential collision. A major requirement for this is, of course, the reliable detection of pedestrians in urban traffic environments. However, this task is rendered particularly difficult by at least the following two facts:

- Pedestrians show a very high variability in shape and color due to physical size, clothing, carried items, etc.
- In urban environments, especially in city centers, pedestrians most often appear in large numbers, e.g. when crossing at a traffic light. This results in many occlusions where the pedestrians are only partly visible.

Despite these difficulties, there are already some encouraging approaches to detect pedestrians, majorly based on camera data (e.g. Leibe et al. [2005]), but also using 2D laser range scanners (Arras et al. [2007]) or both (Spinello et al. [2009b]). However, these systems require a certain minimal size at which the pedestrians are visible in the data, which has the drawback that pedestrians that are far away, as well as children, can not be detected reliably. According to the rule of thumb from theoretical traffic lessons, a car that moves with 50 km per hour needs 40 m to come to a full stop. This is still far from the maximal distance at which pedestrians can be detected with current approaches, using a lens that provides still an acceptable opening angle (above 90 degrees). In this work, we present an approach to detect pedestrians that are up to 50 m away while the lens still provides a wide field of view. The size in which the pedestrians appear in the image is as low as 16 by 20 pixels.

### 4.1.1 Related Work

We already explained the state of the art in pedestrian detection in Chapter 3. We here summarize it and concentrate the references in the area of detecting small pedestrians. In image-based people detection, there mainly exist two kinds of approaches (see (Munder and Gavrila [2006]) for a survey). One uses the analysis of a detection window (Dalal and
4.1. DETECTING PEDESTRIANS AT VERY SMALL SCALES

Figure 4.1: Flowchart of our detection algorithm.

Triggs [2005]) or templates (Gavrila and Philomin [1999]), the other performs a parts-based detection (Felzenszwalb and Huttenlocher [2000], Ioffe and Forsyth [2001]). Leibe et al. [2005] presented an image-based people detector using Implicit Shape Models (ISM) with excellent detection results in crowded scenes. An extension of this method that proposes several enhancements has been already shown in Chapter 3, that explains the works of (Spinello et al. [2008a, 2009b]). In the specific area of small scales pedestrian detection very few works are present. Viola et al. [2003] detect small pedestrians (bigger than the ones detected in this work) including a time integration. Efros et al. [2003] uses optical flow reasoning to detect humans and understand actions. Ferrari et al. [2006] classify contours for detecting simple objects (coffee mugs, animals) in clutter by using an iterative path search among linked segments.

The superpixel method has been introduced by Ren and Malik [2003] using a Normalized Cut criterion (Shi and Malik [2000]) to recursively partition an image using contour and texture cues. Other methods have been proposed to obtain quality superpixel segmentations (Engel et al. [2009], Felzenszwalb and Huttenlocher [2004]).

Far away pedestrians are hard to detect from single images: just few pixels define the shape and textures are washed out by the limited image resolution. Most of the methods present in literature use bags of features approaches, like ISM, or detection window approaches, like Dalal and Triggs [2005]. The first methods are doomed by the lack of interests points, the second kind of techniques need to scroll a detection window through all the image area. Motivated by these reasons, we propose an hybrid method that overcomes these two detection concepts by learning a novel small-scale image features and a pedestrian voting model.

4.1.2 Overview

Our proposed technique uses a supervised machine learning algorithm consisting of the following two major steps:

- **Training** Based on a superpixel segmentation proposed by Felzenszwalb and Huttenlocher [2004] and a computation of the image gradient, segments of strong edge pixels are extracted. From these edge segments $s$, we extract feature vectors based
on a combination of histograms of gradient orientations and the angles that each line segment forms with the horizontal axis. These features are used to train an AdaBoost classifier (Freund and Schapire [1997]). In addition, we store the positive training examples in a codebook together with their displacement vectors with respect to the object centers. This is inspired by the voting scheme of the Implicit Shape Model (ISM) approach (Leibe et al. [2005]).

- **Classification** Again, we compute edge segments and feature vectors. Then we run the classifier and collect all votes for object centers that are cast from edge segments classified as positive. Using mean shift mode estimation (Fukunaga and Hostetler [1975]), we obtain object positions for which many edge segments vote and thus are strong hypotheses for the position of an object, i.e. a pedestrian (see also Figure 4.1).

### 4.1.3 Contributions

Our approach avoids both the necessity of a sliding window, as e.g. in (Dalal and Triggs [2005], Viola et al. [2003]), and the requirement of a minimal number of detected interest points (e.g. Hessian or Harris corners (Mikolajczyk and Schmid [2005])) to obtain robust hypotheses for small objects such as in Leibe et al. [2005], Spinello et al. [2008b]. We present the following novelties:

- the segmentation of edges from the gradient image using a superpixel segmentation. This divides the edges into chunks of homogeneous gradient variability and provides highly informative local edge features. This overcomes the usage of an overlapping tessellation to learn object features and uses a more semantical subdivision: at these image sizes, superpixels tend to segment persons into more meaningful parts like torso, head, limbs. The reason for that is that, due to the smaller resolution, the gradient variability is usually lower than at higher scales.

- a novel image descriptor particularly suited for the detection of objects at small scales,

- a classifier based on a combination of AdaBoost and the voting scheme known from the ISM approach.

### 4.1.4 Feature Extraction

In the literature, many different approaches are presented to compute local interest point detectors and appropriate region descriptors (for a comparison see Mikolajczyk and Schmid [2005]). However, for our particular problem of object detection at very small scales, none of these approaches is well suited for the following reasons:

1. In areas of many small objects, usually – if at all – only few interest points such as Harris corners or Hessian blobs can be detected. Thus, the number of possible voters is very low compared to the number of objects to be detected. This results in
4.1. DETECTING PEDESTRIANS AT VERY SMALL SCALES

detection results with low confidence. We therefore decided to use edges instead of interest points, as described below.

2. Standard descriptors such as SIFT (Lowe [2003]), Histogram of Oriented Gradients (HOG) (Dalal and Triggs [2005]), and shape context (Belongie et al. [2002]) represent the local information in a high dimensional feature space. One could think of applying such descriptors to all (or some) points of an edge chain, but this would result in a large number of feature dimensions. Given that the size of the objects to be detected usually ranges only about 300 pixels, this seems inappropriate.

As a conclusion, we aim at the definition of a simple but informative descriptor that is defined on chains of edge pixels and can be computed efficiently. The decision to use chains of edge pixels or, as we will denote them, edge segments, is somehow inspired by the use of edgelets for detecting pedestrians (see Wu and Nevatia [2005]). In the following, we present the details of our method to compute edge segments and local descriptors.

4.1.5 Superpixel Segmentation

The aim of this first step of our detection algorithm is to preprocess a given input image and to obtain a more semantic representation that is independent on the pixel resolution of the image. One common way to achieve that is by grouping image pixels into regions in such a way that all pixels in a region are similar with respect to some kind of similarity measure. In the literature, this is also known as image segmentation, and it is crucial for a large number of algorithms and applications in computer vision. Many different algorithms have been suggested for this problem and we refer to the related work section in Felzenszwalb and Huttenlocher [2004] for a good overview. Two of the more recent and mostly used approaches, namely Ren and Malik [2003] and Felzenszwalb and Huttenlocher [2004], define a graph where the nodes are the image pixels and the graph edges are defined by a neighbor relationship between pixels. Of these two, the approach by Felzenszwalb and Huttenlocher is more tuned for computational efficiency and the one by Ren and Malik is more robust and yields more informative regions. For our application of detecting small scale objects, the use of complex similarity measures such as the peaks in contour orientation energy as in (Ren and Malik [2003]) is not required. Therefore, we use the former approach in our framework. It defines a weight for each graph edge that corresponds to the dissimilarity measure of the two pixels connected by the edge (e.g. the pixels’ intensity difference). The algorithm then groups the vertices into segments so that the minimal weight of all edges across two segments is still higher than the maximal weights of edges within both segments. As a result, the algorithm produces a set of homogeneous image regions – usually named superpixels, where the number of superpixels (or the level of coarseness) can be adjusted by a parameter $k$, which in principle determines the allowed distance between weights of inter-segment edges and intra-segment edges.

An important characteristic of this method is its ability to preserve details in low-variability image regions while ignoring details in high-variability regions. Therefore, it is especially suited for our application, because pedestrians at small scales are usually represented by only very few pixels in which the color variability is comparably low due
to the lower sampling resolution. This means that one superpixel often represents an entire body part like a leg, a torso, or a head.

4.1.6 Edge Segments and the Edge Descriptor

As mentioned above, we need to find a descriptor that is not only assigned to single interest points, as those occur less frequently in areas of small scale objects. Using superpixels as regions of interest are a much better choice here, as they are always found and they represent a higher vicinity. However, defining a region descriptor for superpixels would result in very complex computations. For our purpose, this is not appropriate, as we only want to represent the information contained in small image regions. As a tradeoff between single pixels and regions, we use edge segments, which are defined as chains of edge pixels that lie inside a superpixel. For the computation of the edge pixels, we apply the Sobel operator to the grayscale image and remove edges with a gradient magnitude that is below a threshold $\tau$. From that, we compute the edge segments by simply applying the superpixel segmentation described above to the edge image.

Adapted to our choice of edge segments we define a descriptor that reflects the local information of each edge segment. This information is later used for our object detection algorithm. In accordance to the notion of a region descriptor, we refer to this as an edge descriptor. In our experiments, we tested the following two kinds of edge descriptors:

- **Histogram of orientations**: The local gradient orientations along an edge segment are collected in a histogram with $n$ bins: each bin $B_i$ counts the number $e_i$ of edge points $p$ at which the gradient $\gamma(p)$ has a certain orientation (see Figure 4.2, left). For the descriptor, we use $2n$ values, where the first $n$ are the values $e_i$, normalized by the sum $m := \sum_{i=1}^{n} e_i$, and the second $n$ values are the sums $\sum_{p \in B_i} |\gamma(p)|$ for each bin $B_i$, again normalized by $m$. We name this descriptor HIST.

- **Vector of directions**: First we compute for each edge segment a polyline approximation consisting of $l$ line segments. We do this using a variant of split-and-merge. Then, we collect all angles between the line segments and the horizontal axis in a vector of length $l$ (see Figure 4.2, right). We name this descriptor VECT.

4.1.7 Feature Classification

Based on the feature extraction described in the previous section, our goal is to formulate an algorithm that classifies these feature vectors into one of the two classes ‘pedestrian’ or ‘background’. For this task, we employ a supervised learning technique that uses a hand-labeled training data set with positive examples of small scale pedestrians. Many techniques have been proposed to achieve this task. Two very successful approaches are the face detection algorithm of Viola et al. [2003] and the voting technique named Implicit Shape Model (ISM) by Leibe et al. [2005]. The advantage of the first method is the strength of AdaBoost (Freund and Schapire [1997]), i.e. a classifier that is arbitrarily accurate on the training data and at the same time yields a rating of the most relevant
4.1. DETECTING PEDESTRIANS AT VERY SMALL SCALES

Figure 4.2: The two types of edge descriptors used in our detection algorithm. Left: Histogram of orientations: for each edge point of a segment we use the orientations of the corresponding gradient, here shown in yellow on four sample edge points, and compute a histogram over them. Right: Vector of line segment orientations: From a polyline approximation to the edge segment, here shown as yellow arrows, we store the orientation angles of each line segment in a vector.

feature dimensions for classification. The downside is that the image has to be searched with a sliding window approach and at different scales. In contrast, the voting scheme of ISM relies on scale invariant features that are stored in a codebook along with the relative position of the object center. No feature search is needed here in the image, but the algorithm does not rank certain feature dimensions over others when finding matches in the codebook. Thus, extracted feature vectors may vote for a potential object center, even though they reveal a low evidence for the occurrence of the object.

In this work, we suggest to combine both ideas to a method that pre-classifies a given feature vector using AdaBoost (see Section 3.6.3.1) and then, in the case a positive classification, searches for a vote of the object center in the codebook.

We designed AdaBoost with decision stumps as weak classifiers. We remark that decision stump finds a hyperplane $\eta$ in feature space that is perpendicular to one feature dimension. It is uniquely defined by the index of the feature dimension, the orientation of the normal vector of $\eta$, and the distance of $\eta$ to the origin.

The features extracted in the previous step are expressed as a $2n$ dimensional point for the first case (HIST) and as a $l$ dimensional point in the second case (VECT). Features quality are evaluated by learning a classifier for each kind of descriptor. Moreover, we measured the quality of the combination of descriptor VECT with HIST concatenating their values in a single feature of dimension $2n + l$. We call this descriptor MIX.

4.1.8 Descriptor Codebook

The main idea of voting based classification techniques, such as the one described by ISM, is to collect a set of image descriptors together with displacement vectors, usually
4.1.9 Detecting Pedestrians

Once the AdaBoost classifier is trained and a codebook is created from the training data, our detection algorithm proceeds as follows. For a given input image, the gradient map and the superpixel segmentation is computed. Using the latter ones, we obtain the edge segments of the test image. Then, we compute the descriptors as described above and classify them with an AdaBoost classifier. All descriptors that are classified positive, are matched to the entries in the codebook. Here, we do a range search to find all descriptors $d$ that are within a given Euclidean distance $r$ from the query descriptor $d_q$. Then, all the votes cast from these descriptors are collected in the voting space by adding their displacements to the centroid of the edge segment for which $d_q$ was computed. In the last step, we apply mean shift mode estimation in the voting space to find the most likely object center for the given votes. Here, we set the kernel radius to half of the width of the
4.1. DETECTING PEDESTRIANS AT VERY SMALL SCALES

training images. To initialize the mean shift estimation, we first collect all votes in a 2D histogram with $0.5w \times 0.5h$ bins where $w$ and $h$ are the width and height of the test image, and then start mean shift at the position of the biggest bins. After convergence of mean shift, we obtain all object hypotheses. From these, we retain those that have a minimum number of votes $\tau_v$.

4.1.10 Experiments

To evaluate our detection algorithm quantitatively, we applied it on a large set of test images with labeled positive and negative examples. We trained our classifier with images from pedestrians in two sizes, namely $16 \times 20$ and $32 \times 40$ pixels. This corresponds in our case to an approximate distance of $56m$ and $28m$, respectively (the focal length of our lens is 4.2mm).

4.1.10.1 Setting the Parameters

As mentioned before, our algorithm depends on several parameters: the superpixel coarseness $k$, the gradient strength threshold $\tau$, the length of the descriptor vectors $m$ and $n$, and the distance parameter $r$ for codebook clustering. To determine these parameters, we created a validation dataset of 2000 random images and evaluated 25 combinations of these parameters on these images. To limit the parameter search space, we chose $m$ and $n$ from the interval $[6, 8]$ as described in Dalal and Triggs [2005], and $\tau$ from $[0, 40]$ to ensure that at most 15% of the gradient information is lost (considering that the maximal possible value is 255). The parameter combination with the maximal sum of true positive and true negative detections was used in the later experiments. We obtained $k = 25$, $m = 8$, $n = 8$, $r = 18$ and $\tau = 25$.

4.1.10.2 Training

For training, we used an internationally standard dataset: the NICTA large pedestrian image dataset of Overett et al. [2008]. It contains pictures with pedestrians taken in typical urban environment. They appear either alone or in crowded scenes with partial occlusions, in different poses (walking, standing, sitting) and in a broad range of lighting variations. Negative examples are represented by random crops of images from indoor and outdoor environments.

We randomly selected 10000 positive images and 50000 negative images for each scale. In total we trained our algorithm with 120000 image samples. In each image we encountered between 10 and 20 edge segments, i.e. several million descriptors were used for training. We used 5 times more negative training examples to provide a large variety of background. To assess the quality of the AdaBoost training we used a leave-one-out cross validation, in which data is partitioned into subsets such that the analysis is initially performed on a single subset, while the other subsets are retained for confirming and validating the initial results. The training was performed on a quad core Intel Xeon CPU with 4GB of RAM in several hours of processing time.
CHAPTER 4. REASONING WITH SINGLE IMAGES

Figure 4.4: Precision-Recall graph for our detection algorithm (SP) and for AdaBoost with Haar features as in Kruppa et al. [2003] (AH). Due to the special design of our descriptor for small scales, the detection on the smaller images yields better results. The green line depicts the Equal Error Rate (EER).

4.1.10.3 Quantitative Results

The test set is composed of 24000 images with 4000 and 20000 negative examples. We evaluated our algorithm for three different kinds of edge descriptors (see Sec. 4.1.6): Histogram of orientations (HIST), Vector of directions (VECT), and both (MIX). The evaluations of the three types of classifiers (HIST, VECT, and MIX) are shown in Table 4.1 and Table 4.3-left for image size 16 × 20 and in Table 4.2 and Table 4.3-right for 32 × 40. The precision-recall values are depicted in Figure 4.4, along with the result from the full-body detector for 14 × 28 images by Kruppa et al. [2003]. This method, outperformed by our technique, uses AdaBoost with Haar features and is very similar to the one that is described by Munder and Gavrila [2006] as close to the best.

The VEC descriptor yields a much lower True Positive Rate (TPR) than the HIST descriptor, which is most probably due to the information loss caused by the polyline approximation of an edge segment. Note that the False Positive Rate (FPR) of both descriptors are similar. The best results are obtained using the combination (MIX) of both descriptors that improves each statistics. It is important to remark that the results for images of size 16 × 20 are generally better than for those of size 32 × 40. The reason for this the specific design of our feature descriptors: a bigger image scale tends to exhibit
4.1. DETECTING PEDESTRIANS AT VERY SMALL SCALES

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Prediction</th>
<th></th>
<th>Ground truth</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>N</td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>70.5%</td>
<td>29.5%</td>
<td>P</td>
<td>56.6%</td>
</tr>
<tr>
<td>N</td>
<td>18.0%</td>
<td>82.0%</td>
<td>N</td>
<td>18.8%</td>
</tr>
</tbody>
</table>

Table 4.1: Confusion matrix for pedestrians of size 16x20 for HIST descriptor (left) and VECT descriptor (right)

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Prediction</th>
<th></th>
<th>Ground truth</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>N</td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>66.7%</td>
<td>33.3%</td>
<td>P</td>
<td>53.0%</td>
</tr>
<tr>
<td>N</td>
<td>23.8%</td>
<td>76.2%</td>
<td>N</td>
<td>23.9%</td>
</tr>
</tbody>
</table>

Table 4.2: Confusion matrix for pedestrians of size 32x40 for HIST descriptor (left) and VECT descriptor (right)

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Prediction</th>
<th></th>
<th>Ground truth</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>N</td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>71.8%</td>
<td>28.2%</td>
<td>P</td>
<td>68.4%</td>
</tr>
<tr>
<td>N</td>
<td>17.1%</td>
<td>82.9%</td>
<td>N</td>
<td>22.1%</td>
</tr>
</tbody>
</table>

Table 4.3: Left: Confusion matrix for pedestrians of size 16 x 20 by using MIX descriptor. Right: Confusion matrix for pedestrians of size 32 x 40 by using MIX descriptor.

a higher level of detail, therefore the superpixel segmentation yields edge segments that are less distinctive compared to those from the low scale images. In the latter ones, superpixels represent body parts at a higher semantic level (legs, heads, arms), whereas at larger scales, the superpixels are less informative. Moreover, due to the fact that low scale images have a lower resolution, mainly strong edge pixels prevail. This means that thresholding the gradient map at the same value $\tau$ results in a lower loss of information. Nevertheless, the proposed technique performs comparably well for images at higher scale: the TPR is only about 3% and the TNR is only about 5% lower.

As a qualitative result, we show in Fig 4.5 the detection result from a fraction of the test data set at image size 16 x 20. All images are arranged in a grid and the estimated object centers are depicted with yellow dots.
Figure 4.5: Qualitative result of our detection algorithm. 600 full size images of correct detections from the test dataset are shown here in matrix form. The yellow dots are the estimated object centers. To keep the presentation uncluttered, the detected bounding box for each image is not displayed.
4.2 Exploiting Repetitive Object Patterns for Model Compression and Healing

Many man-made and natural environments consist of a set of similar basic elements that frequently appear in regular patterns. In this work we present an unsupervised approach for discovering repetitive patterns of objects in a single image. We propose an unsupervised detection technique based on a voting of image descriptors. Then we introduce an object frequency analysis based on latticelets: a minimal set of arcs able to completely describe the connectivity of the graph of detections. Latticelets are then used for building polygonal cycles: the smallest cycles that define a group of repetitions. High level image compression is obtained by representing the image with the median color inside the repetition groups and full color averaged templates extracted from the element class detection. Polygonal cycles are used for high level inference using Conditional Random Fields in order to deduce where repetitive elements can be, even if they are obstructed or detected with low confidence. Our method has been tested on simulated and real data and the quantitative and qualitative result show the effectiveness of the approach.

4.2.1 Contributions

Several environments consist of a set of similar basic elements that appear frequently in a mostly regular pattern. Examples of this include architectural elements such as windows, pillars, arcs, etc., but also other artifacts that are most typical for urban environments such as equi-distant trees, street lamps, or even similar houses that are built with a regular distance apart from each other. Despite the fact that repetitive patterns are ubiquitous, they have got little attention in robotics so far (see Triebel et al. [2007]). There are however, at least two main applications where this information can improve the perception task: occlusion handling and data compression. For the former pattern information can be used to infer the shape and position of an occluded object from similar objects in the same scene. For data compression, knowing the characteristics of a regular repetitive kind of object and its repetition pattern, makes it possible to efficiently represent the environment by only storing the object class description and the pattern. Recently Schindler et al. [2008] used repetitive patterns from images for obtaining geo-positioning.

In this work, we present an automated technique to extract such repetitive patterns and to exploit this information for occlusion handling and compression. We apply our method to the problem of architectural facade analysis, and propose methods to

- complete missing facade information due to occlusions or low detection quality.
- compress the relevant facade information using patterns of repeated detected elements,

However, our proposed algorithm is not restricted to images and can also be applied, for instance, to data that has been acquired with 3D range scanners. The only requirement here is that a method to detect similar objects in a scene must be available.
In this work we present a technique to find repetitive patterns and to represent them with a lattice that is later used in order to detect partially occluded elements. We train a Conditional Random Field that takes into account the repetitive neighborhood information to infer the location of an object even if the detection quality of that object is low. The contributions of this work include:

1. Unsupervised detection of mutually similar objects. Closed contours are extracted and robustly matched in the facade by using a growing codebook approach based on Implicit Shape Models (Leibe et al. [2004]).

2. Analysis of pattern repetition by using \textit{latticlets}: a selected set of frequent distances of the same element in the cartesian plane. With a \textit{latticlet} it is possible to satisfy a complete connectivity among elements of the same class.

3. Higher level analysis of elements repetitions by analyzing the geometrical consistency of cyclical neighborhoods. Conditional Random Fields are employed for inferring occurrences of objects in case of weak hypotheses. Element detection probability and geometrical neighborhood consistency are used as node and edge features.

To the authors’ best knowledge there are not other work that obtain the same goals currently in literature.

\subsection*{4.2.2 Related Work}

In this work we specifically analyze the repetitions from a single static image. The seminal work of Dick et al. [2004] attempted to achieve automatic labeling of architectural elements. We make use of Implicit Shape Models (Leibe et al. [2004]) to detect architectural elements, which is similar to the work of (Mayer and Reznik [2005]) in which a predefined training set for detecting windows is used. In contrast, we do not use any training set, but instead analyze each closed contour and create a codebook of it. This is then refined in order to group similar elements of the facade.

It is important to mention that in other fields, like computer graphics, grammar based procedural modeling (Müller et al. [2007], Wonka et al. [2003], Xiao et al. [2008]) has been formally introduced to describe a procedural modeling of buildings. Most of these works do not aim to discover patterns but just to reconstruct the 3D appearance of the facade and they require human intervention. In this work we automatically aim to describe the grammar of an image, e.g. of a facade.

In the field of facade completion methods based on Markov Chain Monte Carlo (MCMC) (Neal [1993]) have been used, without including information about the connectivity between the detected elements. Approaches based on RANSAC (Schaffalitzky and Zisserman [1999]) and Hough transform (Turina et al. [2001]) have been used to find regular, planar patterns. They often use very specialized models and show examples on a restricted set of images. Moreover, the input needs to be clipped and rectified before any computation. Occluded objects are seldomly take in account, primarily because of stringent assumptions on the nature of symmetric patterns.
4.2. EXPLOITING REPETITIVE PATTERNS

More sophisticated methods relax the assumption of the regular pattern using Near-Regular Textures (NRT) (Hays et al. [2006], Liu et al. [2004]). The latter proposed an iterative algorithm for NRT discovery by higher-order correspondence of visually similar interest points. This method is computationally expensive and it might retrieve tessellations which do not correspond to semantically meaningful patterns. A technique proposed by Ahuja and Todorovic [2007] extracts texels from homogeneous, 2.1D planar texture with partial occlusions. The drawback consist that texels have to be uniformly distributed in the image without any global structure. Bayesian approaches based on Markov Random Fields (MRF) have also been employed for discovering grid structures in genome sequencing (Hartelius and Carstensen [2003]). If repetitive elements are found, Lin and Liu [2006] propose an MRF model to track dynamic deformable lattices. Similar to our work is the one of Korah and Rasmussen [2008], in which the authors propose a method to find repeated patterns in a facade by using NRT with MCMC optimization, using rules of intersection between elements. They focus on extraction of rectangular patterns from which they can extract just a single pattern, based on a 4-connectivity lattice. We are not restricted in generating a single pattern: our method automatically produces and clusters multiple repetition patterns without restriction on the connectivity type. Moreover, we are able to analyze and describe patterns of several object categories in the same image.

It is important to mention two projects, related to the possible uses of the proposed method, that aimed to reconstruct facades of 3D cities: the MIT City Scanning Project (Teller et al. [2003]) and the 3D City Model Generation (Dick et al. [2004]). These works both disregard repetitive patterns topologies and instead they use photogrammetry blending and interpolation methods.

4.2.3 Overview

The first step of our algorithm (see Figure 4.6) is to compute a set of standard descriptors – in our implementation we use Shape Context descriptors (Mikolajczyk and Schmid [2005]) – on a given input image. Then, we compute closed contours on the image. These closed contours represent the candidates for repetitive architectural elements such as windows or pillars. The key idea here is that we do not classify these elements using a model that was previously learned from training data, but instead obtain an evidence of their occurrence by extracting similarities directly from the data of the given scene. The advantage of this is two-fold: first, we are independent of a previously hand-labeled

Figure 4.6: Schematic overview of our proposed algorithm.
training data set and thus of the kinds of objects to be detected. Second, only those objects that are similar to each other are considered to belong to one class with many instances and only for such classes a search for repetitive patterns is meaningful. In other words, classes with only one representative are disregarded for the following steps.

Our measure of mutual similarity is based on the Implicit Shape Model approach introduced by Leibe et al. [2005]. For each object \( O_i \), we create a codebook \( C_i \) of descriptors from inside its contour and use \( C_i \) to match \( O_i \) with all other objects in the scene. We compute a voting score \( q \) that takes into account the spatial coverage of the voting descriptors, similarly to the method explained in Section 3.8.3.9, to obtain element detections. We then define a similarity function between two different object classes \( O_i \) and \( O_j \). Finally we use agglomerative clustering, to group objects that are close to each other under this similarity measure. Each such group we denote as a class \( C \).

In the next step, we analyze the repetitive pattern inside each class \( C \). We do this by looking at the Euclidean distances between objects of the same class in the scene. Using a 2D Generalized Hough Transform, we find relative positions of objects that occur frequently in the image. These relative positions are represented as edges in a lattice graph, where nodes represent objects. The most dominant edges by which all nodes in this graph can be connected are found using the Minimum Spanning Tree algorithm of Kruskal [1956] and grouped into a set that we call latticelet. Using the latticelet, we extract the smallest cycle that occurs in the lattice graph.

Once we obtain the lattice elements and the topology of the smallest cycle in the lattice graph we can predict the position of occluded or not reliably detected object instances by growing the lattice graph beyond its structure as extracted from the input image. Using an inference engine based on Conditional Random Fields we can then determine if the occurrence of an object instance at a predicted position is likely or not. In an image compression application, we can use a template of the detected object class, the medium background color and the lattice structure to efficiently store and retrieve a given input image. All steps of our algorithm will be now described in more detail.

### 4.2.4 Extraction of Mutually-Similar Object Instances

In this section we explain the process of discovering repetitive elements present in an image based on closed contours.

#### 4.2.4.1 Object Extraction

As first step of the algorithm, Shape Context descriptors (Mikolajczyk and Schmid [2005]) are computed at Hessian-Laplace interest points. Then we compute contours using a Canny (Canny [1986]) edge extractor. We refer to the content in each contour as an object instance. Matching contours in real world images can be very hard due to shadows and low contrast areas. We therefore employ an Implicit Shape Model-like (ISM) technique in which the contours act as containers to define a codebook of included descriptors. This way, we can robustly match objects. In summary, an ISM consists of a set of local region descriptors, called the codebook, and a set of displacements, usually named votes,
4.2. EXPLOITING REPETITIVE PATTERNS

For each closed contour, a codebook of descriptors is created that contains relative displacements to the object centers (votes). Then, the descriptors of each object are matched against the descriptors in the image.

for each descriptor (see also Section 3.8.3). The idea is that each descriptor can be found at different positions inside an object and at different scales. Thus, a vote points from the position of the descriptor to the center of the object as it was found in the training data.

In our case all the descriptors found inside the contour are included in the codebook \( C \) as well as the relative displacement of the respective interest points with respect to the center of the contour. To detect repetitions of an object instance in a scene, we match objects to each other in the following way:

1. All descriptors found in the image are matched against an object’s codebook \( C \). Those with a Euclidean distance to the best match in \( C \) that is bigger than a threshold \( \theta_d \) are discarded.

2. The votes that are cast by the matching descriptors are collected in a 2D voting space.

3. We use mean shift mode estimation to find the object center from all votes. This is referred to as an object hypothesis.

To select valid hypotheses we propose a quality function that balances the strength of the votes with their spatial origin. Votes are accumulated in a circular histogram around the hypothetical object center. The detection quality function is given by:

\[
q_i = w_a \cdot \frac{f_h(\alpha_i, \alpha_e)}{f_h(\alpha_e, \alpha_e)} + (1 - w_a) \cdot \frac{s_i}{s_e} \quad q_i \in [0, 1] \tag{4.1}
\]

where \( \alpha_e \) is the vote orientation histogram of the object class \( O_e; \alpha_i \) is the vote orientation histogram of the hypothesis \( i; f_h \) is a function that applies an XOR operator between the bins of two histograms and sums the resulting not empty bins. \( s_i, s_e \) are respectively the score (number of votes received from the hypothesis) and the score of \( O_e \). \( w_a \) is the bias that is introduced between the two members. This is a simplified version of the cost function explained in (Spinello et al. [2009b]). In this work we aim to match elements with the same scale. Detected elements are selected by a simple minimum threshold \( \theta_q \) on the detection quality \( q_i \).
Figure 4.8: Latticelet discovery process. Objects of the same class are detected. A complete graph is built and the relative distances in the cartesian plane are accumulated.

Each detected element of the same class is used to enrich the codebook: descriptors that contributed to each match are added, therefore a more complete description of the variability of the element can be achieved.

4.2.4.2 Refining elements hypotheses

In order to group similar element classes, unsupervised hierarchical agglomerative clustering with average linkage is performed, see Algorithm 3. Element classes are compared to each other based on the similarity of their codebooks. The distance between descriptors of each codebook is computed, and the similarity value between two codebooks is given by

\[ d(C_i, C_j) = L(C_i, C_j) \min \left( |C_i|, |C_j| \right) \]

where \( L \) computes the number of corresponding descriptors from the two codebooks with a Euclidean distance of less than \( \theta_d \) and \( |C_i| \) is the size (number of descriptors) of the codebook \( C_i \).

It is important to notice that, at the end of the matching process, overlap between element classes usually occurs. In order to avoid this problem we run a greedy processing method that assigns descriptors to element classes. Two classes are considered to overlap when their bounding boxes intersect. In each of these intersections we re-assign all the codebook descriptors contributing to the detections to the element class that has more elements, and then we remove those descriptors from the other classes. We then iterate the voting process and intersection check until no more overlapping classes can be found.

4.2.5 Analysis of Repetitive Objects

In this section we explain the procedure to analyze and extract repetitive patterns from sets of discovered object classes.
4.2. EXPLOITING REPETITIVE PATTERNS

4.2.5.1 Latticelets

Here, we introduce the space frequency analysis for the discovered objects. We name the object detection locations in the image as ‘nodes’.

In order to analyze the repetition pattern of each object class we build a complete graph that connects all the centers of the detected objects. Our non trivial aim is to select in this graph edges that have a repeated length and orientation. Moreover we require our arc selection to include all the nodes. Our proposed solution is based on the use of a Minimum Spanning Tree (MST). From the complete graph we build a frequency map (see scheme Figure 4.8 and Figure 4.9), in which we store the distances $dx, dy$ in pixels between nodes of the graph. The map represents the complete distance distribution between the nodes. We therefore have to select from this map the most representative modes. In order to estimate local density maxima in the frequency map we employ a two dimensional mean shift algorithm, with a simple circular kernel. Each convergency mode is expressed by a point in the map $d\hat{x}, d\hat{y}$ and its score repetitiveness that is given by the number of points contributing to the basin of attraction. All the graph edges that contribute to each mode convergency are then labeled with their associated distance. At the end of this process we have obtained a graph in which the distances between the nodes have been relaxed by averaging similar consistent distances/orientations. Moreover, each edge has been tagged with its repetitiveness score.

As last step of this processing we employ Kruskal’s algorithm (Kruskal [1956]) to find the minimum spanning tree by using the nodes, their edge connectivity and the weight of the arcs. The resulting tree represents the most repetitive arcs sufficient to cover all the nodes. In order to compact the information we select each kind of arc just once. We call it latticelet, the minimal set of repetitive arcs that are needed to represent the original lattice. Each object class is therefore associated to a latticelet.
4.2.5.2 Cycles and chains

Latticelets contain very local information, they explain the direction of a possible predicted object from a given position. In order to incorporate higher level knowledge of the repetitive pattern of the neighborhood, we use latticelets. Our aim is to find minimal size repetitive polygons. They provide the effective object repetition that is used in later stages to obtain prediction and simplification (see Section 4.2.6). For each class we sort the weight of its latticelet arcs and we select the one with highest weight. As soon as a lattice is established, we compute the smallest available cycle $\Gamma$ by computing its girth (i.e. length) $\gamma$.

A cycle $\Gamma$ is computed by using an approach based on a Breadth-first Search algorithm. Starting from a node of choice in the graph, arcs are followed once, and nodes are marked with their number of visits. As soon as the number of visit for a node reaches 2 a cycle is found. This is done for all the nodes present in the object class detection set. We then collect all the cycles, and we select the one with the smallest number of nodes. We then create a lattice by using the connectivity offered by $\Gamma_k$ and mark as removed the nodes that are connected by it. Thus, we add another latticelet until all the detection points are covered or all the latticelet are used. We obtained a polygon set composed of frequent displacements (latticelet) that well explains the object distribution in the image (see scheme Figure 4.10). An object class $O_i$ is therefore described by $k$ small cycles: $G_i = \{\Gamma_1, \ldots, \Gamma_k\}$.

Moreover, the algorithm tries to represent with chains the nodes that cannot be described with polygonal cycles. The procedure is analogous to the former one: chain arcs are selected by using the sorted latticelet set. The procedure is run for each $O_i$.

Figure 4.10: From the graph created by an incremental set of latticelet’s arcs, small repetitive cycles $\Gamma$ are selected by using a Breadth-first Search algorithm. Chains are created on the remaining nodes that have not been satisfied by any polygonal cycle $G$.

4.2.6 Structure Inference using Conditional Random Fields

So far, we showed our method to detect objects represented as closed contours and to find repetitive patterns in the occurrence of such objects. However, in many cases, objects can not be detected due to occlusions or low contrast in the image. In general, the problem of these false negative detections can not be solved, as there is simply not enough evidence
4.2. EXPLOITING REPETITIVE PATTERNS

of the occurrence of an object. In our case, however, we can use the information that similar objects have been detected in the same scene and that all objects of the same kind are grouped according to a repetitive pattern. Using these two sources of information, we can infer the existence of an object, even if its detection quality is very low. We achieve this by using a probabilistic model: each possible location of an object of a given type \( \tau \) is represented as a binary random variable \( l_\tau(x) \) which is true if an object of type \( \tau \) occurs at position \( x \) and false otherwise. In general, the state of these random variables can not be observed, i.e. they are hidden, but we can observe a set of features \( z(x) \) at the given position \( x \). The features \( z \) here correspond to the detection quality defined in equation (4.1). The idea now is to find states of all binary variables \( l_\tau = \{l_\tau(x) | x \in X\} \) so that the likelihood \( p(l_\tau | z) \) is maximized. However, in our formulation we will not only reflect the dependence between the variables \( l_\tau \) and \( z \), but also the conditional dependence between variables \( l_\tau(x_1) \) and \( l_\tau(x_2) \) given \( z(x_1) \) and \( z(x_2) \), where \( x_1 \) and \( x_2 \) are positions that are very close to each other. The intuition behind this is that the occurrence probability of an object at position \( x_1 \) is higher if the same object already occurred at position \( x_2 \). We model this conditional dependence by expressing the overall likelihood \( p(l_\tau | z) \) as a Conditional Random Field (CRF) (Lafferty et al. [2001]).

A Conditional Random Field (CRF) is an undirected graphical model that represents the joint conditional probability of a set of hidden variables (in our case \( l_\tau \)) given a set of observations \( z \). A node in the graph represents a hidden variable, and an edge between two nodes reflects the conditional dependence of the two adjacent variables. The theory behind CRFs has been already explained in Section 3.6.3.3.

4.2.6.1 Node and Edge Features

In order to use a CRF we need to define node and edge features, as we have done in 3.13 and 3.15. As mentioned above, the features in our case are directly related to the detection quality obtained from equation (4.1). In particular, we define the node features as \( f_n(q_i, l_\tau) = 1 - l_\tau + (2l_\tau - 1)q_i \), i.e. if the label \( l_\tau \) is 1 for a detected object, we use its detection quality \( q_i \), otherwise we use \( 1 - q_i \). The edge feature function \( f_e \) computes a two-dimensional vector as follows:

\[
\begin{align*}
\frac{1}{\gamma}(f_{e1} f_{e2}) & \quad \text{if } l_{ti} = l_{tj} \\
(0 \ 0) & \quad \text{else}
\end{align*}
\]

with

\[
\begin{align*}
f_{e1} &= \max(f_n(q_i, l_{tj}), f_n(q_j, l_{ti})) \\
f_{e2} &= \max_{G \in G_{ij}} (f_n(\eta(G), l_{ti})),
\end{align*}
\]

where \( G_{ij} \) is the set of (maximal two) minimum cycles \( \Gamma \) that contain the edge between nodes \( i \) and \( j \), and \( \eta(\Gamma) \) is a function that counts the number of detected objects along the cycle \( \Gamma \), i.e. for which the detection quality is above \( \theta_q \).

4.2.6.2 Network Structure

The standard way to apply CRFs to our problem would consist in collecting a large training data set where all objects are labeled by hand and for each object class \( \tau \) a pair of node and edge features is learned so that \( p(l_\tau | z) \) is maximized. However, this approach has two major drawbacks:
• For a given object type $\tau$, there are different kinds of lattice structures in which the objects may appear in the training data. This means that the connectivity of a given object inside its network varies over the training examples. Thus, the importance of the edges over the nodes cannot be estimated in a meaningful way.

• In such a supervised learning approach, only objects of types that are present in the training data can be detected. I.e., if the CRF is trained only on, say, some different kinds of windows, it will be impossible to detect other types of objects that might occur in repetitive patterns in a scene. Our goal however, is to be independent of the object type itself and to infer only the structure of the network. In fact, the object type is already determined by the similarity detection described above.

To address these issues, we propose a different approach. Considering the fact that from the training phase we only obtain a set of node and edge weights $w_n$ and $w_e$, which do not depend on the network geometry but only on its topology, we can artificially generate training instances by setting up networks with a given topology and assigning combinations of low and high detection qualities $q_i$ to the nodes. The advantage of this is that we can create a higher variability of possible situations than seen in real data and thus obtain a higher generalization of the algorithm. The topology we use for training has a girth $\gamma$ of 3 and is shown in Figure 4.11 on the left. Other topologies are possible for training, e.g. using squared or hexagonal cycles, but from experiments we carried out it turns out that the use of such topologies does not increase the classification result. The graph in Figure 4.11 right illustrates that. It shows the true positive and the true negative rates from an experiment with 100 test data sets, each consisting of networks with a total of 5000 to 10000 nodes. The training was done once only with a triangular topology (TriTop) and once also including square and hexagonal topologies (MixTop), which represent all possible regular tessellations of the plane. As the graph shows, there is no significant difference in the two classification results. In contrast to the topology, the number of outgoing edges per node, i.e. the connectivity, has a strong influence on the learned weights. Therefore we use a training instance where all possible connectivities from 2 to 6 are covered as shown in Figure 4.11.

In the inference phase, we create a CRF by growing an initial network. From the analysis of repetitive patterns described above, we obtain the minimal cycle length, the topology and edge lengths of the lattice. By subsequently adding cycles $\mathcal{G}$ to the initial network obtained from the already detected objects, we grow the network beyond its current borders. After each growing step, we run loopy belief propagation to infer the occurrence of objects with low detection quality. The growth of the network is stopped as soon as no new objects are detected in any of the 4 directions from the last inference steps.

4.2.7 Image Compression

One of the aims of this work is to prove that it is possible to compress the information contained in the image (e.g. a facade) by using the proposed detection technique and the space repetition analysis.
4.2. EXPLOITING REPETITIVE PATTERNS

Figure 4.11: Left: Triangular lattice topology used for training the CRF. The numbers inside the nodes show the connectivity of the nodes. Right: Comparison of CRF performances using TriTop and MixTop datasets for training. Shown are the true positive (TP) and the true negative (TN) rates obtained. The result from the TriTop data are shown in box-and-whiskers mode, the MixTop result as dots. From the figure, it is clear that using different topologies for learning gives no significant change in the classification result.

We therefore reduce the image to a simple set of detected object classes, their repetition scheme, and a simplified background extraction. More in detail: each element class is stored as a set of averaged descriptor in the codebook, a rectangular colorscale bitmap resulting from averaging the image areas inside the detected elements bounding boxes.

In order to visually simplify the image background, we assume that the space between detected elements groups is covered by textures of the same kind. In order to have a simplified but similar representation of the foreground we compute different steps:

- We sort element classes groups using their population count.
- We sample squared patches of facade texture in the space between detected elements subgroups. As a texture simplification procedure we compute the average color \( \hat{c}_{RGB} \) of the samples and then the median distance in RGB space with respect to it. This color is used as a simplification of the foreground area. A more sophisticated method can be employed, for example a texture analysis like the one described in (Portilla et al. [2000]), but is far beyond the scope of this work.
- We assign a vertical stripe, that extends from top to the bottom of each detected element group by using previous sorting until all the facade is covered. We fill each stripe with the relative computed color. Empty spaces are filled with the color of the most populated group.

Some examples are shown in the rightmost column of Figure 4.14, 4.15, 4.16.
CHAPTER 4. REASONING WITH SINGLE IMAGES

4.2.8 Experiments

The goal of our experimental evaluation is to investigate to which extent the proposed algorithm is capable to detect different classes of objects, to detect repetition rules and to run inference based on that information.

4.2.8.1 Quantitative results

In order to obtain rich statistics on a wide range of element classes, we generated an evaluation set based on 200 images with high contrast polygons of different size (see Figure 4.12 for an example).

![Figure 4.12: Two samples of the synthetic set used for quantitative evaluation of the algorithm. The picture shows rich repetitive patterns and classes.](image)

Here one of our goals is to finely assess the quality of class distinction and grouping, that is fundamental for the creation of graph analysis. It is important to note that the angle difference between an hexagon and a pentagon is just 12° and in small scales, due to pixel aliasing, this difference may not be easy to be distinguished. Figure 4.13 right shows the average difference between the number of detected classes and annotated classes. The graph is plotted with respect to the minimum detection quality $\theta_b$ needed for each node. We can notice that the algorithm tends to underexplain the data trying to not overfit single detections.

Moreover we want to evaluate the contribution of CRF to the image elements coverage. We again plot, in Figure 4.13 left, this measure with respect to $\theta_b$ and we overlay the results using CRF. When a high coverage is obtained, the CRF contributes with just a moderate performance increase (4%), when just few elements are detected a sound 20% enhancing has been registered. If just a small group of high strength detection elements are obtained, a set of $G$ is created therefore the other aligned elements will be detected. Important to mention is the average of false positives per image: 0.2. CRF therefore increases the true positive rate and it guarantees a very low false positive rate.

4.2.8.2 Qualitative results

A set of images of facades and other repetitive elements have been downloaded from internet and treated as input for our algorithm, as is shown in Figure 4.14, 4.15, 4.16. On each of the examples the difference from discovery and CRF-completed image is shown. It is interesting to notice that the algorithm works also for not rectified facades and several kinds of architectural or repetitive elements. In the scope of this work it is
4.3 Conclusions

In this chapter we have shown how is possible to obtain scene analysis by reasoning on single images. We presented a novel image based detection method for pedestrians at very small scales and an approach for discovering and reasoning on repetitive patterns of objects in a single image.

The first work was published in Spinello et al. [2009a]. For this particular prob-

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig13.png}
\caption{Left: Average difference between the number of detected classes and annotated classes Right: Discovery only detection and discovery + CRF detection. The contribution of CRF is particularly evident when a low detection rate is obtained.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig14.png}
\caption{Left: Discovered elements are depicted with a colored box different for each class. Center: Detection after CRF inference: magenta points are nodes of the expanded lattice, edges are in yellow. The yellow thick line depicts the lattice set $\Gamma$ that well represents the detections. Right: High level compression of the image: simplified foreground and averaged bitmap class is overlayed (for clarity) over the original image.}
\end{figure}
Figure 4.15: Left Column: Extracted self-similar objects (red boxes). Note that often only a few number of instances are found. Center Column: Final CRF lattice (dots and lines) and inferred position of objects (boxes). Right Column: Reconstruction of images based on our high-level compression.

Problem with sparse visual information we propose a new feature descriptor inspired by edgelets in combination with superpixel segmentation. This technique overcomes common drawbacks of the standard interest point voting approach and of the detection window approach by introducing a descriptor codebook and a robust AdaBoost classification technique. We have evaluated parameters and show quantitative results on a large dataset, showing the effectiveness of our method. In future works we want to investigate how an intelligent tracking can improve the results and how to improve the feature robustness with respect to the scale magnification.

The algorithm of the second work is based on the analysis of repetitive patterns. Latticelets are defined by clustering the minimum spanning tree of the complete graph.
4.3. CONCLUSIONS

Figure 4.16: **Left Column**: Extracted self-similar objects (red boxes). Note that often only a few number of instances are found. **Center Column**: Final CRF lattice (dots and lines) and inferred position of objects (boxes). **Right Column**: Reconstruction of images based on our high-level compression.

built over the detection points. Latticelets are then used for building a set of small repetitive cycles. High level image compression is obtained by representing the image with the median color background inside the repetition groups and full color averaged templates extracted from the element class detections. Minimal cycles are used for high level inference using Conditional Random Fields that complete the lattice explaining low strength detections. This method has been tested for quantitative and qualitative result on simulated and real data showing the effectiveness of the approach.
All mobile robots operate in real environments, either indoors, for example in office environments, or outdoors, for example in urban like scenario. With time, more and more robots will participate and share the same environment with humans, like for example autonomous cars or cleaning robots. Sharing the same space means also obtaining that a certain level of safety and interaction must be achieved. In order to create a robot that is intelligently aware of the space around itself, we need to process an analysis of the scene. We need to increase its level of perception of the environment. With this thesis we give a contribution in the field of scene analysis, that is achieved by using a multimodal approach. We focused our work on range and vision data processing. Each of these sensors produce different but complementary kinds of information about the environment for generating region of interest in outdoor, as we discussed in Chapter 2, and for detecting and tracking pedestrians, as we explained in Chapter 3. In certain situations a robot might be forced to give an interpretation to complex situations in which range data is not available, then images can be powerful cues for the scene analysis, specially to detect far away objects, unreachable with lasers, or use repetitive image patterns as a descriptor of the scene (Chapter 4).

5.1 Discussion

This thesis presents a way to perform challenging scene analysis for mobile robots. We here present the conclusions and the motivation of the works presented in this thesis.

5.1.1 Chapter 2: Regions of Interest Generation Using Camera and Laser

A mobile robot in outdoor scenario is often overwhelmed by the amount of sensory information. Tens of information-rich image frames and range data segments are received
each second: processing quickly such amount of data for detecting a moving object or for avoiding an obstacle could be problematic, specially in a crowded scenario. In this chapter we describe two works that deal on reducing the information amount contained in data, in order to focus only on salient regions of interest. We propose multimodal methods based on laser and camera processing.

The first method is based on a fusion of 3D range data and camera images. The aim is to segment areas in which artefacts in natural environments (like a forest or outdoor scenario) could be found. The topic of generating regions of interest for the specific task of detecting man-made structures in natural environments got relatively low attention in the field of robotics. Motivated by this reason we use a multimodal 3D laser and camera setup to create rules of smoothness in 3D geometrical data and color uniformity in image data to obtain a segmentation of the environment by using Bayesian fusion methods. Even though this method is sensitive to point density change in 3D data and lighting conditions, it represents one of the few attempts done in literature in trying to exploit the unordered state of outdoor environments, not as noise but as a valid important cue.

Region segmentation in image sequences has the objective of segmenting areas with similar motion. Motion segmentation can be used to create a dynamic region of interest in which potential objects could be found. The hardest problem is to obtain a generalized method that is effective when the observer is not static and the observed objects are moving. Several methods have been developed in literature to achieve this goal. The drawback of these techniques consists in the requirement of several image frames to produce a solution or the requirement of heavy computational resources. With our work we provide a solution in just two frames by using local rules of similarities between motion vectors of the optical flow. We introduce Markov Random Fields (MRF) that run on Local Entropy values. Moreover, each motion cluster is tracked in the subsequent frame in order to set a prior of motion stability in the following optical flow segmentation. The drawback of the beneficial smoothing process caused by MRF is that boundaries of small clusters of motion would be removed or incorporated into adjacent bigger clusters, if present. In general, this is not a problem for a low speed robotic platform: small motion clusters are associated to far away objects, therefore a low speed mobile platform moving in that direction would perceive increasingly bigger motion clusters. We quantitatively proved this method by using sequences extracted from standard datasets and from data retrieved by a mobile robotic platform.

Both the techniques presented in this chapter are based on probabilistic methods that handle the uncertainty and incompleteness of the information in a real world environment.

5.1.2 Chapter 3: Multimodal Object Detection in Urban Environments

Pedestrians and cars are very important objects in urban environment. Literature shows that several robust methods exist to achieve this goal by using different sensors, among them camera based methods and laser based methods are the most diffuse and successful. A real robotic system that shares the environment with humans have to efficiently function in different crowded situations, in various weather conditions and in various lightning
5.1. DISCUSSION

Camera based methods, even though very robust, have the drawback of needing a sufficient image contrast to work efficiently and of being dependent on the camera-lens for the size of the field of view. Laser rangefinder based methods have the drawback of obtaining just few points per scan, therefore detection is reliably achieved in close range, where point density is high. Lasers function without environment light and in virtually every weather condition. Motivated by these reasons, we overcome the problematics associated to each sensing modality by using a multimodal approach that combines an image based detector with a range based detector. The system is based on an image detector, laser based detector and a sensor fusion step. The image detector has been progressively updated from Histogram of Oriented Gradients to a detector inspired on the Implicit Shape Models, called ISMe. Several contributions regarding feature selection, matching and weighting, subpart reasoning, template shapes and multiclass capabilities have been proposed. Also the laser based detector has been progressively updated in each paper: firstly it was based on AdaBoost classification of Delaunay JDC segments and then on Boosted Conditional Random Fields (CRF). In the first work, sensor fusion is computed by combining the detectors output via Bayesian modeling then a tracker has been developed, in which multiple motion model Kalman filters compete for generating the best hypotheses. The current range of the detector (15m) might be enhanced by using other sensor systems (stereo camera, 3D laser) or by computing an hallucinated position in space of each out-of range detected object in camera by using a scale-distance-angle calibrated system. Most of the ISMe could be heavily parallelizable (feature extraction, mean shift, matching, sorting, multiclass hypothesis selection) and could be implemented on Graphical Processing Units (GPU) of common commercial graphic cards. We evaluated the performance of the system by processing several datasets acquired in real urban environments.

5.1.3 Chapter 4: Reasoning with Single Images

Vision is a sensor modality that has enormous potentials. Sometimes an image is everything a robot could retrieve to infer the solution of a certain problem. In this chapter we propose a solution to the problem of detecting pedestrians at very small scales and a method that focuses in discovering and in using probabilistically the information found in repetitive patterns on single images.

Far away pedestrians are hard to detect from single images: just few pixels define the shape and textures are washed out by the limited image resolution. Most of the methods present in literature use bags of features approaches or detection window approaches. We overcame these standard detection paradigms and we introduced an effective and compact method to detect very small pedestrians in single images. We employ a superpixel clustering to create edge segments from the gradient image. The segments are encoded with custom and compact descriptors that are classified with AdaBoost. Each correctly classified segment vote for the object center like in ISM. A way of using this system might be to use it in combination with the method presented in Chapter 3: when objects disappear because they pass the maximum range limit, the small scale pedestrian detection method is activated and takes care of detecting objects. We quantitatively evaluated, on
standard datasets, that our approach outperforms other standard small scales detectors.

In this chapter we present an ambitious technique to analyze images for repetitive patterns. Repetitive patterns analysis in images has been well studied in literature but little attention has been given to higher level interpretation of patterns, like repetition of windows in a building facade or to the visual sequence of columns on a temple. We fill this void by exploiting the arrangements of ‘high level’ objects found in an image by using geometrical probabilistic inference based on CRF. With this, we can infer predictions to include occurrences of weak detections and to achieve data compression. High level objects are generated automatically by analyzing closed contours. Latticelets of repetitive objects are stored and used for finding repetition cycles and chains. The pattern analysis method is very generic and it could be used for other pattern discovery tasks. Moreover, it could be combined with standard object detector/recognition techniques in order to encode specific arrangements of windows, doors or other architectural elements.

5.2 Future Works

Several future works could spring from this thesis. We here list the most intriguing ones.

5.2.1 Far to close range realtime robust multiclass detection

An interesting future work could be to integrate the presented multiclass multimodal detection and tracking system in realtime on a mobile vehicle, as partially mentioned in Section 5.1.2 and 5.1.3. GPU implementations and parallelization could be used to reach this goal. Moreover, the idea is to integrate small scale detection when objects are far, by formulating hypotheses on their position in front of the vehicle and then use the multimodal approach to detect and track them as soon as possible. A good idea could be also to integrate the optical flow independent motion analysis to help detecting moving objects at far or close range. Such a system can provide a precious and reliable tool for realtime planning and autonomous navigation.

5.2.2 Large Scale Repetition Pattern Analysis

Until now the technique introduced for repetitive pattern analysis, explained in Section 4.2, has been applied on images of facades or on other architectural elements. An intriguing future work could be to employ it in large scale, specially in aerial imagery to create compact descriptors of cities or neighborhoods. If we combine this information with other local properties (e.g. wealthiness, criminality, parking space, number of trees) this could produce a precious tool for city planners, governmental agencies and city councils.
Auntter: A Mobile Urban Platform

In the scope of this thesis we participated in the development of an advanced mobile platform, named Auntter, for outdoor and urban mobile robotics. The aim of the platform is to research new methods for navigation, mapping and scene analysis. The robot is based on a standard Smart GmbH (formerly MCC Smart GmbH) car, an automotive brand of Daimler AG. The car has been equipped with distance laser sensors, cameras, a differential GPS unit, an Inertial Measurement Unit (IMU), an optical gyroscope and several computers. Computers communicate the information provided by the various software modules and drivers via an efficient and centralized inter process communication system.

A.1 Vehicle description

The vehicle model is a Smart ForTwo Coupé Passion produced in 2005 equipped with a 45kW engine. This model has been chosen because it is compact, light, has power steering, automatic gear shift, and a CAN-bus that is easily accessible. All these features facilitate the process of converting the vehicle for autonomous driving.

A 24V power generator has been mounted in order to power all the electronic devices and additional actuators. The generator is driven by a belt and pulley system that is directly connected to the engine output axis. Two batteries placed in the trunk act as an energy buffer. They have a total capacity of 48Ah and are continuously recharged when the engine is running.

A specific electronic board has been designed in order to use the power steering motor for drive by wire. To fulfil this goal, the CAN-bus of the vehicle is disconnected from the electronics of the car and routed to a computer so that the steering angle is accessible to our controller.
A system of cable and pulleys is used to activate the break pedal. The motor that pulls the cable is placed under the driver seat and is commanded by the CAN computer. Moreover, a specific electronic system has been designed to enable the usage of a computer to set the gas command.

### A.2 Sensors

Sensing is a fundamental part in mobile robotics. The modular design of the sensors supports allows an easy reconfiguration of the sensor placements. In this way we can prepare the car for custom experiments or data logging (see Figure A.1). The following are the sensors most commonly mounted in the car and used in this thesis:

- **Laser line scanning**: Sick LMS291-S05, outdoor version, waterproof. *Ibeo Alasca XT* multi layer laser scanner.
- **3D laser**: two Sick LMS291-S05 lasers vertically arranged on top of a continuously rotating turntable. For each revolution, at 1Hz frequency, a full 3D scan of the environment is acquired.
- **Inertial Measurement Unit**: Crossbow NAV420, waterproof. A 3 axes accelerometer, 3 axis gyroscope, a 3D magnetic field sensor and a GPS receiver (not used).
- **Omnidirectional camera**: Sony XCD-SX910CR camera, lens focal length 12mm, with a parabolic mirror.
- **Monocular cameras**: Sony XCD-SX910CR or Marlin F-033C with exchangeable lenses: wide angle lens (focal length 4.2mm) or tele lens (focal length 8mm)
- **Differential GPS receiver**: Omnistar Furgo 8300HP, equipped with an external rain proof antenna.

### A.3 Multimodal Sensors Configurations used

In the scope of these thesis, for producing multimodal data logging and conducting experiments based on camera and laser, we used different sensors configuration. The different setups are listed in table A.1 and visually shown in Figure A.2.

### A.4 Software architecture

The Smartter computing power is mainly based on 5 computers based on linux operating systems. They are placed in the trunk of the car and they are connected together via a Gigabit Ethernet hub. Most of the software has been developed in C++ programming language by using open source software and libraries. *BASH* scripting is also used for automatizing tasks or ease the software launching procedures. Robotics software often
A.4. SOFTWARE ARCHITECTURE

Figure A.1: A typical experimental session with the mobile robotic platform Smartter.

requires more than one source of information from more than one sensor or software module, therefore a process communication system has been used. The Carmen Robotics Toolkit, introduced by Montemerlo et al. [2003], is the robotic software architecture selected for this task. We specially employed its encapsulation of the Inter Process Communication protocol (IPC) for communicating data through custom defined messages between different software modules and sensors in the network. The Carmen architecture has also the possibility of easily log all the timestamped IPC messages managed. This allows an easy offline replay of a data retrieval session, that is a great help for software development and research related tasks.

Special care has been taken when logging high bandwidth streams from firewire cameras. A special threaded software module reduces the overhead of the operating system by saving consecutive images in huge file chunks (named RAWTAR format). For this reason a software suite for visualizing, extracting and converting has been implemented. With this system the problematic issue of frame dropping when retrieving a video stream is virtually eliminated.
APPENDIX A. SMARTTER: A MOBILE URBAN PLATFORM

<table>
<thead>
<tr>
<th>Vision system</th>
<th>Range system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conf 1</td>
<td>Ibeo Alasca XT set at 15Hz.</td>
</tr>
<tr>
<td>A Sony XCD-SX910CR, wide angle lens, 15fps, fixed behind the wind-screen, centered, close to the top border</td>
<td></td>
</tr>
</tbody>
</table>

| Conf 2        | Ibeo Alasca XT set at 15Hz and 3D laser at 1Hz |
| A Sony XCD-SX910CR, wide angle lens, 15fps, fixed behind the wind-screen, centered, close to the top border |

| Conf 3        | Ibeo Alasca XT set at 15Hz. |
| A Martin F-033C, tele lens, 15fps, placed on the rooftop on the left side |

Table A.1: Multimodal configuration, laser and camera, used in the experiments.

Figure A.2: Laser and camera configurations (see table A.1). The blue rectangle indicates the position of the laser scanner, the red circle the position of the camera and the green square the 3D laser.
Curriculum Vitae

A.5 Biography

Luciano Spinello was born on 11 February 1980 in Rome, Italy.

In May 2005, he received his Laurea degree (equivalent to a M.S.) in Automation Engineering at the University of Roma TorVergata. He got the final grade of 110/110 Summa cum Laude.

The title of his Master dissertation was: 
Solving the SLAM problem through particle filters and extended Kalman filters on mobile robotics localization and map building. His Master thesis supervisor was Dr. Francesco Martinelli.

From November 2005 to January 2006, he did an internship in Autonomous Systems Lab (ASL) under the supervision of Prof. R. Siegwart at the Swiss Federal Institute of Technology of Lausanne (EPFL).

From January 2006 to present, he has been PhD candidate at the D-MAVT department of the ETH under the supervision of Prof. Siegwart.

Since 2006 he has been member of the IEEE community.

A.6 Awards

Full marks cum laude engineering graduation, 2005

Second place for the "Best Innovative Thesis" award, an Italian nationwide award for Master Thesis, 2006
A.7 Academic Projects


AxtCamera MATLAB toolbox for calibrating laser-scanners and cameras. 2008

Initiator and organizer of the "ASL Tutorials". A regular reading group for mobile robotics with participants from ETH and guest speakers. 2008-2009

Deep renewal of electrical, software, hardware system of robot Nomad XR400 owned by University "Tor Vergata" Rome. Production of the only existent complete documentation of hardware and software of the robot: “Guida al Nomad XR4000”. 2004

A.8 Academic Activities

First author of eight papers published in major robotics and artificial intelligence conferences (ICRA, AAAI, IROS, FSR, IFAC)

Author of four workshop papers presented to major robotics and artificial intelligence conferences (ICRA, IROS)

Invited talk at "People Detection and Tracking Workshop", ICRA 2009

Session co-chair in the regular session "Intelligent Vehicles & Intelligent Transportation Systems" for the IROS 2008 conference, Nice - France

Reviews for many conferences and journals

A.8.1 List of publications and workshops


A.8. ACADEMIC ACTIVITIES


Bibliography


