MONOGRAFIES DE L'INSTITUT D'INVESTIGACIÓ
EN INTEL·LIGÈNCIA ARTIFICIAL
Number 36

Institut d'Investigació
en Intel·ligència Artificial

Consell Superior
d'Investigacions Científiques

Localization and Object Recognition for Mobile Robots

Arnau Ramisa

Foreword by Ramon Lopez de Mantaras

2010 Consell Superior d'Investigacions Científiques
Institut d'Investigació en Intel·ligència Artificial
Bellaterra, Catalonia, Spain.

Series Editor
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Foreword by
Ramon Lopez de Mantaras
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Volume Author
Arnau Ramisa
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Institut d'Investigació
en Intel·ligència Artificial

Consell Superior
d'Investigacions Científiques

NIPO: 472-10-154-0
ISBN: 978-84-00-09150-7
Dip. Legal: B.45998-2010

Ordering Information: Text orders should be addressed to the Library of the IIIA, Institut d'Investigació en Intel·ligència Artificial, Campus de la Universitat Autònoma de Barcelona, 08193 Bellaterra, Barcelona, Spain.

al Mati i la Nuri

Contents

List of Figures

Foreword

Endowing robots with reasoning capabilities allowing them not to just answer the question "where am I?" but also giving them the ability to infer, for example, the type of room they are in, through semantic recognition of the objects, persons,... in the room, is an extremely difficult research goal. Arnau Ramisa, in this PhD thesis, has addressed these issues and the results he has obtained are a step towards this goal. The thesis has two parts. The first one addresses the localization problem and the second one addresses the object recognition problem. The main contribution of the first part is the obtention of a signature to characterize places relying on a constellation of combinations of different types of visual feature regions extracted from a panoramic image of the place that is represented by a node of a topological map graph. The proposed method has been tested with panoramic images of different rooms in various UAB Campus buildings. The results show that through the signature obtained the robot can compute its location even if some changes in the environment have occurred. The second contribution of the first part of this thesis is the development of a biologically-inspired and computationally cheap homing method based on the Average Landmark Vector (ALV). Normally, artificial landmarks had been used together with the Average Landmark Vector, but this thesis introduces a method that combines the ALV with the feature regions used for global localization extracted from the panoramic images mentioned above. Although the localization method proposed in ths first part of the thesis is able to reliably model and recognize places, a powerful space representation constructed from semantically rich elements is needed. To progress towards this goal, the second part of this thesis presents and evaluation of two state of the art object recognition methods - SIFT and Vocabulary Tree - and, more importantly, several improvements of these methods are proposed in order to adapt them to the more strict computational requirements of mobile robots. Furthermore, since the evaluation has shown that the suitability of an object recognition method depends on some specific properties of the acquired images, a Reinforcement Learning based approach has been proposed to select, on-line, which object recognition method should be applied in each situation. This learning approach constitutes the last contribution of this thesis.

Robotic vision is a very challenging and difficult subject. Arnau has successfully achieved several important contributions towards the resolution of fundamental

issues in this field, with a high dose of imagination, creativity, autonomy, and solid scientific and technical background. These are essential qualities to be successful in scientific research. I have been very lucky having Arnau Ramisa as my PhD student because I have personally enjoyed and learned very much working with him. I hope the readers of this monograph will also appreciate the quality and depth of this work.

Bellaterra, October 2010

Ramon Lopez de Mantaras
Director of the IIIA-CSIC

Chapter 1

Introduction

If some day robots are to get closer to what science fiction depicts, they will definitely require a rich perception and representation of the environment. In the past, robots used sonars as instruments for this task. Navigating with a sonar is similar to walking in a dark room trying to feel the walls and objects with the hands. Comprehensibly the first localization algorithms relied almost completely in the movements of the robot, just using the perception of the environment to correct the errors in the odometry (Elfes, 1989, 1990; Moravec, 1988). Later, ladars provided more reliable measurements, however, ladars are still too expensive for most applications, and the 2D range scanners are not able to provide enough information to qualitatively characterize places, not to mention objects. Lately, advances in computer vision along with an improvement in digital cameras have risen an increasing interest within the robotics field towards a vision-based autonomous robot.

However, extracting meaningful information from images has proven to be an arduous task. Myriad problems plague visual information, like perspective transformations introduced when the point of view changes, occlusions or motion blur, just to name a few.

Nevertheless, we humans, as vision-based autonomous navigating agents, get around these problems and manage to recognize places and objects. Numerous studies have been dedicated to understand how animals construct their mental maps of the environment and how they use them to navigate. A notable example is the work of Tolman (1948) where the author introduces the idea of a cognitive map based on ethological experiments with rodents. According to this study, rats construct mental representations of places based on the spatial relation of environment's features.

This theory gained strength when O'Keefe and Dostrovsky (1971) identified *place cells* in rodent brains. This kind of cells are neurons, mainly located in the hippocampus, that fire when the rat is located in certain places. These neurons are activated primarily by visual cues, but also by movements because they show activity even in the dark.

When published, this study had little impact on the robot navigation research

community, still in its first stages. However, time has passed and the field of robot navigation has seen enormous progress. Since then, many interesting robot navigation models taking inspiration from Tolman's theory have been proposed, such as the TOUR model by Kuipers (1978), that is designed as a psychological model of human common-sense knowledge of large-scale space; the RPLAN by Kortenkamp (1993), a model of human cognitive mapping adapted to robotics, and evaluated in an indoor scenario combining inputs from vision and sonar sensors; or the schematic maps by Freksa et al (2000) that, based on the idea of cognitive maps, distinguished between different levels of abstraction for map representation and their applications: representations closer to the geometric reality of the world are useful for local navigation and obstacle avoidance. On the other hand, more abstract or schematic representations can help in global localization and path-planning tasks.

In the beginning, these type of approaches materialized in the form of topological localization systems such as the ones described by Filliat and Meyer (2003), that make an extensive survey of internal representations of spatial layout for mobile robots with a focus on localization, or the more recent topological approach by Tapus and Siegwart (2006), that propose to use a signature which they call *fingerprint* to represent a room. This signature is constituted by a circular string that encodes the distribution of color blobs and sharp edges –extracted from omnidirectional images and a pair of 2D laser range scanners pointing in opposite directions respectively.

Recently, more ambitious approaches in the cognitive sense have been undertaken, as the one by Vasudevan et al (2007), that proposed a hierarchical probabilistic concept-oriented representation of space, constructed from objects detected in the environment and their spatial relationships. Such representation allows to endow the robot with a reasoning capacity that transcends the question "*where am I?*", typically pursued by the previous localization systems, by giving it the ability to infer the purpose or category of the room through the semantically meaningful elements or objects that can be detected.

However, if this type of approaches are to succeed, they will undoubtedly require much more advanced perception capacities than the ones typically found in a robot nowadays. Along the way to this ambitious goals in robotics research, this thesis contains the contributions described in the following section.

1.1 Contributions

The main contribution of the first part of this thesis is a signature to characterize places similar in spirit to the method proposed by Tapus and Siegwart (2006) in that it uses an omnidirectional vision sensor to perceive the environment. However, instead of relying also in range information provided by laser range scanners, the place model proposed here is purely vision based. In this approach, a place is characterized as a *constellation* of combinations of different types of visual feature regions extracted from a panoramic image that can be used as a node of a topological map graph.

These types of features are designed to be resistant to viewpoint and illumination changes and, in consequence, the proposed signature is also resistant to some extent to these problems. Furthermore, as the signature is composed from many individual features, it can tolerate some degree of dynamic changes in the environment while still being capable of recognizing the place. In order to test the proposed method, we have performed localization tests in various sequences of panoramas taken in different rooms of various buildings.
Since analyzing the whole map can become a time consuming task, we propose and evaluate a fast re-ranking method based in the *bag of features* approach to speed-up this step. Finally, we show that the presented signature is notably resistant even in the case of using a conventional perspective camera to perform localization.
In order to allow the robot to move between the nodes of the topological map, the second contribution of this first part is a biologically inspired inexpensive visual homing method based on the Average Landmark Vector (ALV). This homing method is able to determine the direction home by comparing the distributions of landmarks corresponding to the *home* with the current omnidirectional images, but without having to explicitly put them in correspondence. Typically, artificial landmarks have been used in experiments with the ALV. However, the method presented in this work combines the ALV with the feature regions employed earlier for global localization, thus complementing the localization method. First, a theoretical study is performed to evaluate the applicability of the proposed method in the domain of the local features and, next, it is evaluated in real world experiments showing promising results.

Even though the localization method proposed in the first part of this thesis is able to reliably model and recognize places, still few semantic knowledge about the world is available to the robot to reason with. As mentioned earlier, Vasudevan et al (2007) proposed a powerful space representation constructed from semantically rich elements of the environment. However, in order for this model to be applicable, a fast and robust object recognition or classification method is indispensable.
Indeed, not only localization would benefit from having a robust, generalistic and easily trainable object recognition system. Also other fields such as robot manipulation, human robot interaction and, in general, any discipline that addresses a practical use of robotics in a not highly structured environment would benefit from such a method. On the other side, computer vision is obtaining impressive results with recent object recognition and classification methods, but we are aware of little effort on porting it to the robotics domain. Therefore, a lightweight object perception method which allows robots to interact with the environment in a human cognitive level is still lacking.
In order to help reduce a bit this gap, in the second part of the thesis the main contribution is the evaluation of two successful state of the art object recognition methods – the SIFT object recognition method from Lowe (2004), and the Nister and Stewenius (2006) Vocabulary Tree – on a realistic mobile robotics sce-

nario, that includes many of the typical problems that will be encountered when roboticists try to use these methods on practical matters. Both methods have several properties that make them attractive for the problem of mobile robotics: the SIFT object recognition method detects object hypothesis location up to an affine transformation and has a low ratio of false positives; the Vocabulary Tree is a *bag of features* type method that was designed with the objective of being fast and scalable. Furthermore, it is suitable for types of objects that may confuse the SIFT method because of few texture or repetitive patterns. Additionally, and more importantly, several modifications and improvements of the original methods are proposed in this thesis in order to adapt them to the domain of mobile robotics.
The selected algorithms are evaluated under different perspectives, as for example:

- Detection: Does the method have the ability to easily and accurately detect where in the image is located the detected object? In most situations, large portions of the image are occupied by background texture that introduce unwanted information which may confuse the object recognition method.
- Classification: A highly desirable capability for an object detection method is to be able to generalize and recognize previously unseen instances of a particular class, is this achievable by the method?
- Occlusions: Usually a clear shot of the object to recognize will not be available to the robot. An object recognition method must be able to deal with only partial information of the object.
- Image quality: Since we are interested in mobile robots, motion blur needs to be taken into account.
- Scale: Does the method recognize the objects over a wide range of scales?
- Texture: Objects with a rich texture are typically easier to recognize than those only defined by its shape and color. However, both types of objects are equally important and we want to evaluate the behavior of each method in front of them.
- Repetitive patterns: Some objects, such as a chessboard, present repetitive patterns which cause problems in methods that have a data association stage.
- Training set resolution: Large images generate more features at different scales that are undoubtedly useful for object recognition. However, if training images have a resolution much higher than test images descriptor distributions may become too different.
- Input features: Most modern object recognition methods work with local features instead of raw image pixels. There are two reasons for this: in the first place, concentrating on the informative parts of the image the size of

the redundant input data is significantly reduced and, on the second place, the method is insensitive to small pixel intensity variations due to noise in the pixels, as well as small changes in point of view, scale or illumination. We evaluate several state of the art visual feature detectors.

- Run-Time: One of the most important limitations of the scenario we are considering is the computation time. We want to measure the frame-rate at which comparable implementations of each method can work.

From the obtained results, conclusions on the methods viability for the mobile robots domain are extracted and some ways to improve them are suggested. The final aim of this work is to develop or adapt an object recognition method that is suitable to be incorporated in a mobile robot and used in common indoor environments.
However, as it was found that none of the evaluated object recognition methods completely fulfilled the requirements of a robotics application, we proposed, as a last contribution, a Reinforcement Learning based approach to select on-line which object recognition schema should be used in a given image. The Reinforcement Learning approach is based on low level features computed directly from the image pixels, such as mean gray-level value or image entropy. It was evaluated in a challenging dataset and found to have very good performance. Another possible use of this method –although not directly addressed on this work because of time constrains– could be speeding up the Nister and Stewenius Vocabulary Tree by quickly discarding irrelevant areas of an image.

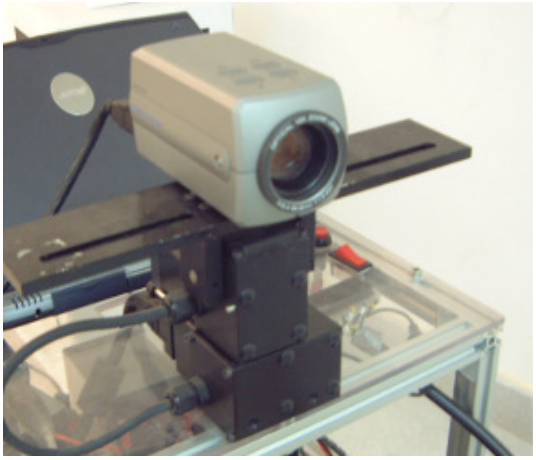
1.2 Robot

All experimentation carried on through this thesis have been done with the help of a real robot. Figure 1.1 shows pictures of it with the single-camera and the stereo head modes. The robot is a Pioneer 2AT, and is equipped with a Directed Perception PTU 46-70 pan tilt unit. On top of the pan tilt unit one or two Sony DFW-VL500 cameras are mounted. The cameras have a resolution of 640×480 pixels. The CPU of the robot is an Acer Travelmate C110 laptop (Intel Pentium M 1000MHz, 799 MHz, 760 MB RAM) placed on top of the robot and running Microsoft Windows XP. For the object recognition experiments, the platform that supports the cameras and the laptop was raised in order to gain a more human-like perspective and be able to see objects on top of the tables.

1.3 Publications

From the work carried on while pursuing this thesis, several publications have been derived:

- A. Ramisa, A. Tapus, R. Lopez de Mantaras, R. Toledo; "Mobile Robot Localization using Panoramic Vision and Combination of Local Feature



(a)

(b)

Figure 1.1: (a) The Pioneer 2AT robot used in the experiments described in Chapters 3 and 4. (b) The stereo setup has been used in Chapters 5 and 6 of the thesis.

Region Detectors", In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, Pasadena, California, May 19-23, 2008, pp. 538-543.

- R. Bianchi, A. Ramisa, R. Lopez de Mantaras; "Learning to select Object Recognition Methods for Autonomous Mobile Robots", In *Proceedings of the 18th European Conference on Artificial Intelligence*, Patras, Greece, July 21-25, 2008, pp. 927-928.

- R. Bianchi, A. Ramisa, R. Lopez de Mantaras; "Automatic Selection of Object Recognition Methods using Reinforcement Learning", In *Recent Advances in Machine Learning (dedicated to the memory of Prof. Ryszard S. Michalski)*. Springer Studies in Computational Inteligence. To appear.

- A. Ramisa, S. Vasudevan, D. Scharamuzza, R. Lopez de Mantaras, R. Siegwart; "A Tale of Two Object Recognition Methods for Mobile Robots", In *Proceedings of the 6th International Conference on Computer Vision Systems, Lecture Notes in Computer Science 5008*, Santorini, Greece, May 12-15, 2008, pp. 353-362.

- A. Ribes, A. Ramisa, R. Toledo, R. Lopez de Mantaras; "Object-based Place Recognition for Mobile Robots Using Panoramas", In *Proceedings of the 11th International Conference of the ACIA, Frontiers in Artificial Intelligence and Applications, Vol. 184. IOS Press*, Sant Marti d'Empuries, Girona, October 22-24, 2008, pp. 388-397.

- A. Ramisa, R. Lopez de Mantaras, D. Aldavert, R. Toledo; "Comparing Combinations of Feature Regions for Panoramic VSLAM", In *Proceedings of the 4th International Conference on Informatics in Control, Automation and Robotics*, Angers, France, May 2007.

- M. Vinyals , A. Ramisa, R. Toledo; "An Evaluation of an Object Recognition Schema Using Multiple Region Detectors", In *Proceedings of the 10th International Conference of the ACIA. Frontiers in Artificial Intelligence and Applications, Vol. 163. IOS Press*, Sant Julia de Lria, Andorra, October 2007, pp. 213-222.

- A. Goldhoorn, A. Ramisa, R. Lopez de Mantaras, R. Toledo; "Using the Average Landmark Vector Method for Robot Homing", In *Proceedings of the 10th International Conference of the ACIA. Frontiers in Artificial Intelligence and Applications, Vol. 163. IOS Press*, Sant Julia de Lria, Andorra, October 2007, pp. 331-338.

- D. Aldavert, A. Ramisa, R. Toledo; "Wide Baseline Stereo Matching Using Voting Schemas", In *1st CVC Research and Development Workshop*, October 2006.

- A. Ramisa, D. Aldavert, R. Toledo; "A Panorama Based Localization System", In *1st CVC Research and Development Workshop*, October 2006.

Besides, three more papers have been submitted for publication:

- A. Ramisa, A. Tapus, D. Aldavert, R. Toledo, R. Lopez de Mantaras; "Robust Vision-Based Localization using Combinations of Local Feature Regions Detectors".
- A. Goldhorn, A. Ramisa, D. Aldavert, R. Toledo, R. Lopez de Mantaras; "Combining Invariant Features and the ALV Homing Method for Autonomous Robot Navigation based on Panoramas".
- D. Aldavert, A. Ramisa, R. Toledo, R. Lopez de Mantaras; "Visual Registration Method for a Low Cost Robot".

1.4 Outline of the Thesis

This thesis contains eight chapters that can be grouped in two parts of closely related content: chapters three and four describe an approach to visual-based indoor global localization without any semantic capability, while chapters five to seven address the issue of object recognition, the main difficulty if a semantically enhanced approach is to be attempted. Chapters two and eight present the related work and preliminaries, and the conclusions and future work respectively. Next is a brief outline of the thesis starting at chapter two.

Chapter 2: Related Work and Preliminaries

In this chapter, we review literature related to both of the robot localization and object recognition fields. Approaches to global localization with similarities to the one proposed are discussed, and interesting methods for object recognition that have some characteristics relevant for robotic applications are presented. Finally, some preliminaries on the type of visual features employed through all this work are reviewed.

Chapter 3: Global Localization Method

In this chapter our proposed topological indoor localization system is presented and evaluated in a dataset of panorama sequences from various buildings. Additionally, a re-ranking of the map nodes to speed up the search of the current location is proposed. Also experiments with conventional perspective images instead of panoramas are performed.

Chapter 4: Appearance-Based Homing with the Average Landmark Vector

In order to travel between the topological map nodes proposed in the previous chapter, here we experiment with the ALV homing method using image features as the ones employed for localization. Simulation experiments, as well as real-world ones, are done to verify the robustness of the method.

Chapter 5: SIFT Object Recognition Method

In this chapter the SIFT object recognition method, as well as the proposed modifications, are described and evaluated. First we review the effects of varying the different parameters of the algorithm and, next, the most successful configurations are further evaluated on the whole test data.

Chapter 6: Vocabulary Tree Method

Similarly to the previous chapter, here different choices of parameters for the Vocabulary Tree method, and the proposed adaptations to detect objects in unsegmented images, are compared. The best performing combination is again evaluated on the whole test data.

Chapter 7: Object Recognition Method Selection with Reinforcement Learning

In our experiments, we found that both methods have advantages and drawbacks and therefore, in this chapter, we propose a Reinforcement Learning approach for selecting the best performing method for a given input image.

Chapter 8: Conclusions and Future Work

Finally, in this chapter, the conclusions of the thesis and future research directions are presented.

Chapter 2

Related Work and Preliminaries

Since the beginning of the 90's one of the best solutions to the robot navigation problem has been *Simultaneous Localization And Mapping* (or simply SLAM) techniques. These techniques consist of a statistical framework to simultaneously resolve the problems of localization and map building combining the information from the odometry and the sensors of the robot. SLAM was proposed in a series of seminal papers by Smith and Cheeseman (1986) and Smith et al (1990).
Traditionally there are two main approaches to SLAM: metrical and topological. In short, metrical SLAM builds a geometric map of the environmentis and recovers the exact position of the robot, while topological SLAM methods build qualitative maps which contain the connectivity between fuzzy-defined places and are particularly useful in path-planning tasks. Between this two main approaches, there is a range of hybrid approaches that combine some characteristics of both paradigms to compensate for the defects of each single approach (Tomatis et al, 2002). Next we give a brief explanation of both paradigms, with their advantages and limitations, and some examples.

Metrical SLAM techniques build maps that reflect the real geometric properties of the environment and are the more frequent SLAM methods. The earliest approach to SLAM used *occupancy grid* maps, first proposed by Elfes (1990, 1989) and Moravec (1988) in the late 80s, a short time before the introduction of SLAM by Smith, Self and Cheeseman. The map representations divide the environment in a high-resolution grid, in which each cell contains the probability of being occupied by obstacles or not. These methods allow the use of raw sensor data without any feature extraction process, but they require a lot of memory. The original motivation for this kind of maps is that the predominant sensor at the time of its development was the sonar, a short-range and mostly imprecise detector.
The main advantages of metric SLAM are that it is simple to reuse a map in

different robots, its construction is straightforward (grid maps consist only of occupied/free cells, and in feature-based maps features are located at their estimated position) and that the map is close to a CAD model (Thrun, 1998). In addition, metric SLAM methods facilitate tasks such as local navigation and obstacle avoidance. However, the most important drawbacks of metric SLAM methods is that they are very space-consuming, no efficient path-planning strategies can be carried on metric maps, a very precise and reliable sensor is required and, in spite of the robustness introduced, there is still a significant dependence on the odometry of the robot.

Kalman filters are the base of the classical SLAM method. The maps in this approach are usually represented by the Cartesian coordinates of sets of features, which can be shapes or distinctive objects in the environment. Using these features and the estimated position and orientation of the robot arranged in a state vector, posteriors are computed with the Kalman filter, along with uncertainties in the landmarks and the robot positions.

The basic assumption taken in Kalman filter mapping is that the motion and perception models are linear with added Gaussian noise. To overcome the non-linearities inherent in the real world, Taylor expansions are used. This modified approach is known as extended Kalman filters. The principal advantage of SLAM based on Kalman filters is that a full posterior probabilities map is estimated in real-time, and its most important drawback is the Gaussian noise assumption, which implies that it can not handle the correspondence problem (Thrun, 2002).

An extension of the Kalman filters paradigm is the algorithm by Lu and Milios (1997) which combines a first phase where a posterior map is computed using Kalman filters with an iterative *maximum likelihood* data association step. The main advantage of this modification over the original method is that it can cope with the correspondence problem as long as the posterior map computed in the first step is accurate enough. Its main drawback is the iterative nature of the method, which makes it not suitable for real-time.

Another extension of this paradigm which obtained very good results uses a Rao-Blackwellized particle filter to maintain various Kalman filters with different hypothesis of the robot position. This way the correspondence problem can be solved by keeping the most feasible hypothesis (Doucet et al, 2000; Thrun, 2002)

Expectation Maximization algorithms for robot localization are based on the work in maximum likelihood by Dempster et al (1977). This SLAM technique performs hill climbing in all the possible maps to find the most coherent one. It consists of two iterative steps: the E-step where the robot position is estimated based on the best available map, and the M-step, which estimates a maximum likelihood map based on the locations computed at the E-step. The principal advantage of this method is that it solves the correspondence problem by repeatedly re-localizing the map relative to the present map at the E-step. With the multiple hypotheses tested, different correspondences are tried and the most ones likely are used. The main drawback of the algorithm is that it is iterative, so it can not be used in real-time and is not incremental. Examples of Expectation Maximization SLAM methods include Burgard et al (1999) and Thrun

et al (1998).
The sensors used in most of the metric SLAM algorithms are traditionally sonars and 2D laser range scanners because they directly retrieve depth information. Therefore the inclusion of a new landmark in its estimated 3D position on the map is straightforward. In recent years several methods capable of performing metric SLAM with purely visual information have started to appear. However, in images the landmark depth information is not directly accessible, so landmark map location must be obtained by reconstruction from two or more views (Hartley and Zisserman, 2004).
One way to classify these visual SLAM methods is by the type of image acquisition hardware used. The *MonoSLAM* method by Davison et al (2007) is the first successful approach to SLAM using a single perspective camera. A map is constructed recovering the position of salient features – computed with the detector of Shi and Tomasi (1994) – using structure from motion. Another approach is to use two or more cameras and obtain the depth of the detected features by means of stereopsis. This approach is the one taken by Se et al (2001), where the 3D position of SIFT (Scale Invarian Feature Transform) features (Lowe, 2004) is estimated with the help of a *Tryclops* trinocular camera system. Finally, an approach that has attracted much attention recently consists in using an omnidirectional camera. With a wider field of view, more features can be detected and matched to the map, resulting in a more robust estimation of the robot position (Murillo et al, 2007). Furthermore, having omnidirectional vision means that potentially every frame acquired with the vision system will have sufficient features for localization. In opposition, perspective cameras can easily be pointed towards an area without enough texture (e.g. a white wall) making localization impossible.

Topological SLAM ignores the geometry of the environment and represents it by a set of descriptions of significant places. The map is usually represented as a graph, where the nodes are the places and the accessibility information between them are the edges. Topological localization methods do not attempt to find the precise position of the robot (a (x, y) coordinate to some reference frame), but to qualitatively know where in the map is the robot. Although different strategies can be used, the most common way for topological localization systems to characterize places is constructing a descriptor based in features of the environment (Filliat and Meyer, 2003).
The most important advantages of topological navigation are the following: it does not require a metric sensor, the qualitative representation of the space can be easily used in high-level tasks (for example path-planning), an objective representation of the environment is not necessary, it requires less memory than a metric map and in general has a lower complexity level. The drawbacks are that most pure topological maps do not suffice for local navigation and that the decision of when and how to update the nodes of the map and when to add new ones is not always clear. In addition, if the localization method does not use odometry at all, then the perceptual aliasing problem gets worse as no movement

information is available to help in the disambiguation of places, such as when closing a loop.

The concept of space representation of topological localization has a strong relation to the way animals construct their mental maps of the environment. As mentioned in Chapter 1, already in the late 40s, E. Tolman, in the light of ethological experiments with rodents, hypothesized the existence of certain parts of the brain associated with certain places the animal had visited and activated through sensory information (Tolman, 1948). He called them cognitive maps.

O'Keefe and Dostrovsky (1971) identified the *place cells* in rodent brains. Place cells are neurons located mainly in the hippocampus that are activated when the rat is in a certain place. The cause of activity of this neurons is primarily sensorial information –in particular visual features– but also movement information is used, as they continue activating in the dark. This evidence poses a stimulating example of a successful topological navigation system. In contrast, Thrun (2002) states that the main difference between metric and topological SLAM approaches is the resolution of the map: in metric SLAM it is fine-grained and in topological SLAM it is coarser. In fact, until recently, most of the topological SLAM algorithms were hybrid approaches where nodes are given by partitioning the free space under some criteria, but no significant qualitative differences between the nodes are present (Tomatis et al, 2002; Choset and Nagatani, 2001; Nieto et al, 2004). This lack of pure topological methods is because SLAM algorithms were originally designed for sonars, and later for 2D laser range-finders. This kind of sensors give a very accurate estimation of depth, but provide none or very few information about the appearance of the environment and, thus, most of the time it is not possible to discriminate one place from another, so no pure topological approaches were possible using that type of sensors.

The situation began to change with the introduction of digital cameras as sensors for SLAM methods. Cameras offer a much richer source of information about the environment's appearance that can be used to characterize places in a distinctive way. An illustrative example of such type of topological SLAM method is Tapus and Siegwart (2006). This method describes places using *fingerprints*. Fingerprints are circular ordered lists of features extracted from the readings of an omnidirectional camera and two 2D laser range-finders. The extracted features are: color patches and vertical edges from the omnidirectional images and corners from the laser range-finders. In addition, the features are enriched with orientation information and long empty spaces are also taken into account. The list of features is then represented as a string, where each character encodes a distinct feature. The match between fingerprints is done using a modified string matching algorithm to cope with the uncertainty introduced by occlusions or changes in the point of view.

Finally, the simultaneous topological localization and mapping process is performed by a partially observable Markov decision process, that again copes with uncertainty to find in which place of the map the robot is and know when to add new information (Tapus and Siegwart, 2006). This approach has been tested in indoor as well as in outdoor environments, with very good results in both. In

addition, loop-closing tests were performed, with successful results.

2.1 Localization

SLAM is composed of two main activities: Localization and Mapping. Localization takes care of deciding where in the map is the robot located and Mapping is about when and how the map must be updated. Although both are important problems, in this work we have addressed primarily the former.
As stated by Thrun et al (2000) three different localization problems are distinguished in the literature:

1. Position Tracking. This type of localization consists in correcting small odometry errors as the robot moves with the help of perceptual information. The uncertainty in the robot localization is usually local in this type of problem, making unimodal state estimators such as Kalman Filters a good option. A similar concept is *visual odometry*, where the movement of the robot is estimated from visual information (Scaramuzza and Siegwart, 2008).

2. Global Localization. In this type of problem the initial position of the robot is not known, and it can't be used to constrain the area of the map where the robot can be located. Consequently the method must be able to robustly distinguish between the different possible positions.

3. Robot Kidnapping (Engelson, 1994). This problem consists of suddenly teleporting a well-localized robot to a random position of the map without letting the robot know. This type of problem simulates a gross localization error and evaluates the ability of the method to recover from it.

In Chapter 3 a localization method is described that is specially suitable for the second and third type of localization. Recently, other robot global localization methods similar to the one proposed in this work have been presented.
Murillo et al (2007) propose a hierarchical method for localization using omnidirectional images. This method uses vertical lines (radial in the omnidirectional image) characterized using color descriptors. The hierarchical nature of the method can deal with large databases by applying three filters of increasing complexity (from linear to quadratic in the number of features) to reject unlikely images. When at least five correct line matches have been computed between the query image and two related images from the dataset, metric localization can be performed using the 1D radial trifocal tensor and a robust model fitting method such as RANSAC. The method has been evaluated in two image datasets both for topological and metric localization with very good results.
Cummins and Newman (2007) propose an appearance-only SLAM system that learns generative models of places to compute the probability that two visually similar images were generated in the same location. The approach is based on the *bag of visual features* type of method proposed by Sivic and Zisserman

(2003). Another example is the approach proposed also by Cummins and Newman (2008), that uses a probabilistic bail-out condition based on concentration inequalities. They have applied the bail-out test to the appearance-only SLAM system and extensively tested their method in outdoor environments.
Furthermore, the work presented by Angeli et al (2008) describes a new approach for global localization and loop detection based on the bag of words method.
Booij et al (2007) build first an appearance graph from a set of training omnidirectional images recorded during exploration. The Differences of Gaussians (DoG) feature detector and the SIFT descriptor are used to find matches between images in the same manner as described in (Lowe, 2004), and the essential matrix relating every two images is computed with the 4-point algorithm (with planar motion assumption) and RANSAC. The similarity measure between each pair of nodes of the map is the ratio between the inliers according to the essential matrix and the lowest number of features found in the two images. Appearance based navigation is performed by first localizing the robot in the map with a newly acquired image and then using Dijkstra's algorithm to find a path to the destination. Several navigation runs are successfully completed in an indoor environment even with occlusions caused by people walking close to the robot.
Valgren and Lilienthal (2008) evaluate an approach focusing on visual outdoor localization over seasons using spherical images taken with a high resolution omnidirectional camera. Then, Upright Speeded Up Robust Features (U-SURF) (Bay et al, 2008), that are not invariant to rotation, are used to find matches between the images and the 4-point algorithm is used to compute the essential matrix. In a previous experiment, the authors found that the performance using these feature regions was similar to the DoG regions described with SIFT (Lowe, 2004) but much more efficient to compute. In order to reduce the unaffordable computational cost of comparing a novel view with every single image stored in the memory of the robot, incremental spectral clustering is used to group together images from a room, or area, that share a similar appearance and find a group representative. Then the matching has a first phase where the novel image is compared to the cluster representatives and, if there is no clear winner, the query image is compared to all images belonging to the putative clusters.

2.2 Towards Semantic SLAM

With the aforementioned techniques impressive achievements have been obtained. However, these purely navigational approaches are not useful for the high-level reasoning we expect from robots that have to assist us in our everyday life. For example inferring the *type of room* the robot is in, or where should it start looking for a milk bottle are tasks that can be only accomplished with a more semantically rich representation of space.
To address this problem, a straightforward solution could be enhance an existing metric map with detected objects and other semantic information. The work of Galindo et al (2005) takes this approach and represents space as a occupancy grid from which a hierarchical topological map is constructed to ease

path-planning and reasoning tasks. Semantic concepts are represented in a parallel hierarchy inferred from the objects detected and *anchored* to nodes of the hierarchical spatial map. Alternatively, Vasudevan (2008) proposed a more integrated approach where the higher-level features with semantic content are the building blocks from which a hierarchical probabilistic concept oriented representation of space is created and grown. This approach brings together the two previously described SLAM types (as in an hybrid approach) and incorporates a new *semantic* layer on top of them. This semantic layer is built from high-level features with semantic content detected in the environment, such as objects or doors instead of corners and lines. The detected objects are inserted in metric object graph maps that constitute the lower level of the space representation. These object graph maps can be used in a similar way as a conventional feature-based metric map. However, its true power is that they can be generalized in a hierarchical way to high-level concepts that represent particular categories of rooms or areas. Several approaches of increasing complexity for the generalization of place categories are presented. The tests included data acquired from 19 living and working environments. Furthermore, experiments on a real plataform with real sensor data where also conducted using a version of the SIFT Object Recognition method presented by Lowe (2004).

2.3 Visual Object Recognition

As can be seen in the recently published literature, currently there is a big push towards semantics and higher level cognitive capabilities in robotics research. One central requirement towards these capabilities is being able to identify higher level features like objects, doors etc. in perceptual data. These high-level features can be perceived using a variety of sensing devices.
Using pure range data for object recognition seems attractive. However 2D laser range scanners, the most typically used sensor for robot mapping and navigation, provides an amount of information that is definitely not sufficient to identify objects, furthermore it is restricted to a single plane. 3D lidars provide a richer source of information. However, its price and working conditions make them still difficult to use in mobile robots.
Recently, a novel type of range cameras that uses infrared light time-of-flight has appeared in the market. These cameras are a convenient way to acquire a 3D image in an indoor scenario. Furthermore, they are compact (in the order of a few centimeters per side) and have a good frame-rate. However this technology is still at early stages, and current models of this type of cameras can only obtain images with a resolution of less than two hundred pixels per side of scenes with a maximum depth of five meters. In spite of the novelty of this technology, research for object recognition with these devices is already being conducted (Gächter et al, 2008).
To date, the most common approaches to object recognition are done using conventional visible spectrum camera images. The major drawback of this type of sensors is that depth is not directly accessible and, if required, it must be esti-

mated using stereopsis, structure from motion or a similar technique. However cameras have a wealth of other advantages, such as much more resolution or an affordable price. Although impressive results are obtained by modern object recognition and classification methods, still a lightweight object perception method which allows them to interact with the environment in a human cognitive level is lacking. Furthermore, the system should be able to learn new objects in an easy, and preferably automatic, way.
Recently methods have been proposed that are quite successful in particular instances of the general object classification problem, such as detecting frontal faces or cars (Viola and Jones, 2001) , or in datasets that concentrate on a particular issue (e.g. classification in the scale-normalized Caltech-101 dataset). However in more challenging datasets, like the detection competition of the Pascal VOC 2007, the methods presented achieved a lower average precision[1]. This low performance is not surprising, since object recognition in real scenes is one of the most challenging problems in computer vision (Pinto et al, 2008). The visual appearance of objects can change enormously due to viewpoint variation, occlusions, illumination changes or sensor noise. Furthermore, objects are not presented alone to the vision system, but they are immersed in an environment with other elements, which clutter the scene and make recognition more complicated. In a mobile robotics scenario a new challenge is added to the list: computational complexity. In a dynamic world, information about the objects in the scene can become obsolete even before it is ready to be used if the recognition algorithm is not fast enough.
Next we review some of the most relevant state-of-the-art object detection and classification methods. Also, we discuss which sources of information can be exploited in the mobile robotics domain in order to have a fast and robust object recognition method, and which of the existing techniques could be applied in such domain.
For clarity, in the following we will refer to *classification* as the task of assigning previously unseen object instances to given general class label (is this a mug?), *recognition* as the task of identifying a particular object instance (is this my mug?) and *detection* as (roughly) deciding which area of the image is occupied by our object of interest (where is the mug?).

Recently, significant work has been done in visual object classification, with many methods making analogies to document retrieval literature. Visual vocabularies (see Section 2.4 for the basics to understand the following related work) have been extensively used to relate local feature regions to visual words by clustering its descriptors with algorithms like k-means (Sivic and Zisserman, 2003; Csurka et al, 2004) or hierarchical k-means (Nister and Stewenius, 2006) among others. This reduces its dimensionality to a fixed number, so visual-word statistics can be used to classify the input image.
Sivic and Zisserman (2003) use k-means to cluster local feature descriptors into a vocabulary of visual words and the TF-IDF (Term Frequency - Inverse Document

[1] `http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/`

Frequency) scheme to prioritize distinctive visual words. Local features are detected in an image with an affine-invariant adaptation of Harris corner detector (Mikolajczyk and Schmid, 2002) and the MSER detector (Matas et al, 2004) and a histogram is built from visual-word counts. Experiments are done in scene matching - where query descriptor vectors are compared to the ones in database - and object retrieval throughout a movie, where a user-specified query region is used to re-rank the results using spatial consistency, accomplished by matching the region visual-words to the retrieved frames. However, creating and using large linear visual vocabularies (millions of visual words) can become an intractable problem. Nister and Stewenius (2006) use the same TF-IDF scoring scheme but this time the vocabulary is constructed using a hierarchical k-means. This allows an efficient lookup of visual words in logarithmic time, enabling the use of larger vocabularies. The results show how well the system scales to large databases of images in terms of search speed, and the larger vocabulary describes much better the images so, in contrast with (Sivic and Zisserman, 2003), geometric information is not really needed to obtain good performance. New query vocabulary trees are compared to the training set contents using k-Nearest Neighbors, therefore the training set is organized in an *inverted files* structure to speed-up the comparison. Jegou et al (2007) highlight that although the inverted files structure speeds up the search, it is still linear in the number of training images, and propose to organize the database images in a two-level structure. The first level consists in an inverted files structure of medoids (the most central element of a cluster used as its representative) computed with the k-medoids algorithm (Kaufman and Rousseeuw, 1990) over the visual vocabulary vectors. The second level contains an inverted file for the vectors assigned to each of the k medoids. This two level structure allows to search only the inverted files for the medoids closer to a query vector. Another contribution of the work of Jegou et al (2007) is a contextual dissimilarity measure to iteratively modify the neighborhood structure to reduce the impact of too-often-selected images.

This concept of *bag of words* approach has been maturing in recent years, with works such as the one by Zhang et al (2007), where multiple approaches (e.g. different alternatives to construct the image signature, various distance measures, etc.) as well as a wide range of parameters are rigorously tested in multiple object and texture datasets. The classifier used in this work is a SVM, using both the *Earth Distance Measure* and the χ^2 distance.

Nowak et al (2006) evaluate different parameter settings for a bag of features approach and propose to use descriptors computed randomly in a dense grid over the image rather than only at positions returned by an invariant feature detector. Even though performance is lower when using few descriptors (feature detectors find more informative image regions) when the number of sampled regions is high enough, their brute force approach can improve the results of feature detectors.

A somewhat related approach is the one by Agarwal and Triggs (2008). They propose a novel representation, which call *hyperfeature* (see Figure 2.1), that is optimized for capturing and coding local descriptors and their co-occurrence

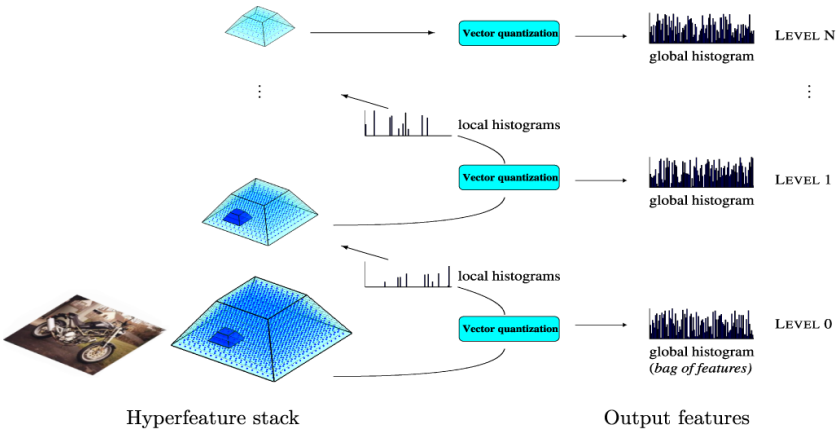


Figure 2.1: Hyperfeature stack. The base level is constructed from descriptors calculated in a dense grid over the image. Subsequent levels are generated from local histograms of codebook memberships from the previous level. This image has been taken from (Agarwal and Triggs, 2008).

statistics. It consists of a hierarchy of bag of features representations. For the first level bag of features, local region descriptors computed over a dense grid are used, as done by Nowak et al (2006). The input of the next level are local histograms of visual word counts from the previous level, treated as if they were local feature descriptors in a standard bag of features approach. Each level of the pyramid is also constructed over scale space to capture objects at different scales. Finally, Latent Dirichlet Allocation is used to reduce the dimensionality and reduce fragmentation of the codebooks.
Serre et al (2007) propose a feedforward approach for object classification, deeply inspired in psychological and biological research, that mimics the primate visual pathway. It has been designed following evidence collected in numerous studies done with monkeys. The model consists of four layers of computational units, alternating simple cells (called computational units) layers with complex cells. In short:

- S_1: The first layer computes the response of the image pixels to a bank of Gabor filters at different orientations and scales.
- C_1: The second layer performs a local MAX operation at each position of a grid defined over the results of the first layer.
- S_2: The third layer compares the results of the previous layer with a vocabulary of approximately 1000 patches in C_1 format on local neighborhoods, acting as Gaussian RBFs.
- C_2: Finally, the last layer takes the maximum over all S_2 associated with each patch, resulting on a vector of the same size of the vocabulary. This vector is then used in a conventional classifier such as SVM or boosting.

The proposed approach is tested on a handful of datasets and for different tasks such as classification, presence/absence detection in unsegmented images or detection with sliding windows. Furthermore, the proposed feature detector is shown to outperform several other detectors/descriptors – such as DoG/SIFT (Lowe, 2004) – and object classification methods like the Implicit Shape Model from Leibe et al (2004). One of the interesting points of this approach is its biological plausibility, another is the fact that being a feedforward model it is easily parallelizable, and therefore with the appropriate hardware can possibly be used in real-time scenarios. However, in its current state the approach stands only for the primary visual cortex –the *what* in the *what/where* ventral/dorsal visual pathways dichotomy (Zeki, 2001)–, and the rest of the visual system of the brain is emulated by a simple linear classifier. Adding the top-down capabilities and feedback connections of the rest of the visual system constitutes an extremely interesting, yet challenging, line of research that aims to yield a vision system similar to biological ones. In (Serre et al, 2006) the system is shown to be comparable to human performance in rapid animal-non animal categorization task. In this task humans are asked to decide if an animal is present in a picture exposed from 30 to 60 ms. This short exposition gives time only to the primary visual cortex to process the information.

The work of Opelt et al (2006a) proposes combining multiple feature types in a single model making use of boosting to form each classifier. This enables each category to choose the best features, regardless of type, that describes it. Local features used are both discontinuity and homogeneity regions. The former are appearance features detected and described with various methods that capture local intensity changes and the later are regions extracted with wavelets or segmentation that contain repetitive textures or stable intensity distributions respectively. Results show that different feature types perform much better in some classes than others, so combining them in a single model greatly improves classification results.

Although good results have been obtained using *bag of words* type methods, when a signifficant ammount of clutter is present in the image or detection and pose estimation is necessary, information on the relative position between object features is essential. This is usually a requirement when dealing with real world images, and several methods have been developed in order to take advantage of this positional information.

Fergus et al (2003) present a generative model to learn object classes as a flexible constellation of parts. The parts are found using the local scale invariant feature detector proposed by Kadir et al (2004), that concentrates on high entropy regions of the image. The different parameters of the objects (appearance, shape, relative scale and occlusions and statistics of the feature finder) are learn jointly in a Bayesian framework. One of the main drawbacks of this approach is the complexity of this combined estimated step which restricts the method to use only approximately 10 parts per object.

Leibe et al (2004) present the Implicit Shape Model (ISM), a method that combines detection and segmentation in a probabilistic framework. First a codebook is built using agglomerative clustering over a set of 25x25 pixel patches computed around Harris corner points. For every resulting cluster center, the Normalized Grey-scale Correlation distance is computed to all the training patches and, for those that are over a certain threshold, the relative object center position is stored. For detection, a 2D Generalized Hough Transform (with continuous space to avoid discretization) is used to cluster probabilistic votes for object centers. Finally, the image is segmented assigning to each pixel the most probable object class with respect to the matched patches that include it. A Minimal Description Length procedure is used to reject overlapping hypotheses based on segmentation results. Tests have been done with the UIUC car database and with the individual frames of a walking cows video database, reporting very good performance. However, results show that the method is able only to deal with very small scale changes (10% to 15%). To avoid this, authors suggest using scale-invariant interest point detectors or rescaled versions of the codebook.

Opelt et al (2006b) present the Boundary-Fragment Model (BFM). This strategy is similar to the one of Leibe et al (2004), but instead of local patches, it uses boundary fragments. A codebook of fragments for detecting a particular class is built by first computing Canny edges of the training images and finding edge segments that match well in a validation set (and bad in the negative examples set) using an optimization procedure. Next the candidate edge segments are clustered using agglomerative clustering on medoids to reduce redundancy, storing also the centroid of the object. Groups of two or three segments that estimate well the centroid of the object are used as weak detectors in a boosting framework. A strong detector is trained selecting weak detectors with good centroid prediction capabilities in positive images and that do not fire in negative images. For detection, weak detectors are matched to image Canny edges, with each one voting for one or various centroid positions in a 2D Hough voting space. Votes are accumulated on a circular window around candidate points, taking those above a threshold as object instances. Finally approximate segmentation can be obtained backprojecting the segments that voted for a centroid back into the image. In order to make the method robust to scale and in-plane rotation, different scaled and rotated of the codebook are used simultaneously.

The authors extended this method for multi-class object detection (Opelt et al, 2006c) using a shared codebook. The method can be trained both jointly or incrementally. Yu et al (2007) propose a shape-based method for object detection. The method builds on the ISM and uses k-Adjacent Segments (Ferrari et al, 2008) with k=3, called Three Adjacent Segments (TAS) as shape-based feature detector. A codebook of TAS is generated by first building a fully connected graph of all the TAS in the training set with distance in the edges between every pair of TAS and then applying Normalized Graph Cuts to obtain the cluster centers. Similarly to the ISM model, probabilistic votes are casted in a 2D Hough Transform, and Parzen Windows are used to find the most plausible object centers. Finally Gradient Vector Flow is used to find an approximate segmentation

of the objects. The proposed method is interesting and has a similar performance to BFM in the cows and motorbikes datasets used by Opelt et al (2006b) although using simpler features, specially in the case of few training images and small codebooks ($\sim$ 200 features).

Lowe (1999, 2004) proposes a single-view object recognition method along with the well known SIFT features. First SIFT features of the training images are compared using Euclidean distance to the set of descriptors from the test image using a K-D tree and the Best Bin First algorithm to speed up the matching process. Matches in which the distance ratio between the first and second nearest neighbors is greater than 0.8 are rejected. This eliminates 90% of the false matches while discarding less than 5% of the correct matches. Matches remaining after this pruning stage are clustered using its geometrical information in a 4D Generalized Hough Transformation with broad bin sizes, which gives the initial set of object hypotheses. To avoid the problem of boundary effects in bin assignment, each keypoint match votes for the 2 closest bins in each dimension. Although imprecise, this step generates a number of initial coherent hypotheses and removes a notable portion of the outliers that could potentially confuse more precise but also more sensitive methods. Bins of the Hough Transform containing three or more matches constitute an object location hypothesis. Finally a least-squares method is used to estimate the affine transformation of the detected object.

The SIFT object recognition method has been also extended to handle 3D objects. Lowe (2001) proposes to create a model from a set of training images taken at different points of the view sphere to gain robustness to viewpoint changes. The model is constructed in an unsupervised manner by clustering together similar training images and establishing links between different views of the same feature. When voting in the Hough Transform for a test image, matches to different adjacent images of the training set are propagated using these links to ensure that at least one model view accumulates votes for all the matching features. Brown and Lowe (2005) use a view-centered approach to find relations between an unordered set of 2D views of an object and, next, all views are combined in an object-centered model. Gordon and Lowe (2006) go one step further and use a 3D object model in an augmented reality task. First, a 3D model of the object is built in an offline stage from a collection of views. Feature points are extracted and matches between each related pair of views are established using camera geometry information to filter false correspondences. Next, matched SIFT features are added to a 3D object-centered model. During the online stage, the pose of the object is estimated by first performing a 2D to 3D feature matching between the query image and the 3D object model, and then using RANSAC and the Levenberg-Marquardt algorithm (Levenberg, 1944) to determine the transformation and filter the outliers.

Bag of features type object classification methods that make no use of geometric information can still be used for detection by taking a sliding window approach. Fulkerson et al (2008) propose several ideas in this direction: to accelerate the evaluation of the windows they use integral images defined over the feature dic-

tionary space. This would still not be sufficient to achieve frame-rate speed in classification if a large dictionary is used, which has repeatedly been shown to improve the performance in recent work (Nister and Stewenius, 2006; Philbin et al, 2007). In order to improve the ratio between dictionary size and classification accuracy, agglomerative information bottleneck (AIB) is used: First large dictionaries are created with hierarchical k-means and next AIB is used to build an agglomerative tree from the leaf nodes creating a coarse-to-fine-to-coarse architecture. This reduces the dictionary size and overcomes the problem of excessively large dictionaries while retaining the performance. The method is tested in the Graz-02 dataset achieving results comparable to those of Marszalek and Schmid (2007) and with an execution time of around 2 seconds. Lampert et al (2008) propose, as a much faster alternative to sliding windows, a branch and bound scheme to rapidly explore the space of all possible bounding boxes for the object of interest in the image. The bounding box parameter space is defined by intervals of rectangles of interest coordinates. Namely, $[T, B, L, R]$ (top, bottom, left and right respectively) defines a rectangle interval with $T = [t_{max}, t_{low}]$, etc. Consequently, the parameter space is defined as a tuple of such intervals. The key element that makes it possible to use branch and bound in this schema is the definition of a quality bound (i.e. maximum score that can be obtained for a given bounding box parameter interval). Using this quality bound, search can be prioritized to sets of parameter intervals that attain the best maximum score. Lampert et al. apply the proposed approach to optimize localization with a simple bag of features approach, with spatial pyramid kernels and even with multi-image search.

Context has been argued to provide very relevant information in computer vision. In Opelt et al (2006a), weak classifiers are selected with a boosting method to form the final classifiers; it was found in the experimental results that a high percentage of weak classifiers with higher weights selected local features corresponding to background. This can be interpreted as that context in some classes is more discriminative than foreground parts, although it greatly depends on the training set used. Also, context can be introduced in an ontology-based manner between objects as it is done in Rabinovich et al (2007), where object co-occurrence probability is computed by extracting related terms to each class from Google Sets or directly computing it from the ground truth. The system first segments query images into regions, which are in turn classified in a bag of features fashion and results obtained are re-ranked based in object co-occurrences. In Torralba et al (2003) context is provided using global features. These are computed from local features extracted with a collection of steerable filters applied over the image, then taking the average magnitude of the responses over a coarse spatial grid in order to obtain a global descriptor of 384 dimensions, which is finally projected to its 80 principal components. Knowing that video frames captured in the same place have a strong correlation, the authors use a Hidden Markov Model to compute the probabilities of being in a specific place given global features from some frames ago. Place recognition results are also

proven to provide a strong prior for object detection.

We are aware of few works that consider the case of general object recognition in the domain of mobile robots. In Ekvall et al (2006) the authors integrate object recognition and SLAM. The proposed object recognition method works with few training images, and consists of two stages: Hypotheses generation and verification. Active vision is used to focus on interesting parts of the image. The first stage uses Receptive Field Coocurrence Histograms (RFCH) and, in the verification stage, an area of interest that contains a maximal number of hypotheses is zoomed in using the optical zoom of the camera. Again RFCH are used to verify each hypothesis and finally SIFT features are extracted and matched to the model image. No further geometrical constraints are aplied to discard outliers and, if an object dependent number of SIFT features are matched, the object is considered detected. Background substraction is used during the training stage to precisely segment the objects. In Murphy-Chutorian et al (2005) the authors present a system that learns and recognizes objects in a semi-autonomous fashion. Their system acquires learning samples of new objects by analyzing a video sequence of a teacher showing an object in a variety of poses and scales. Then it automatically segments and builds an internal representation with a shared vocabulary of visual features. The vocabulary is made using k-means over features based on Gabor-jets and hue-saturation values extracted at Harris corner points. Each visual word stores the relative position of the objects where it appears. Recognition is done by accumulating votes on a Hough Transform in a similar way as in the work of Leibe et al (2004). In the experiments, the system runs at about 1-2Hz, recognizing objects from a 100 object database with 78% of mean accuracy Additionally, the database included difficult objects, like untextured, flexible and very similar objects (wires, cans, water bottles). Pose and scale invariance is obtained by superimposing multiple learning samples, so it is only invariant in the range that the object was learned. Wersing et al (2008) propose a biologically motivated object recognition that runs at 1-2Hz in a 2.4 GHz QuadCore Intel processor. Furthermore, an online learning framework is proposed that uses stereo depth map segmentation to determine interest regions corresponding to the object to train in frames of a video sequence and collects sufficiently dissimilar exemplars for training in a short time memory. These cropped images are used to train View-Tuned Units (VTU) linear discriminating units that respond on a receptive field of shape features and coarse color input. VTUs are trained at three different resolutions to attain good activation levels at the whole size range at which the object may appear in future images. A complete map of VTUs for all objects and scales is then defined covering the whole input image. From this map of responses a model can be created to determine object positions and scale. The model is trained from the responses of the VTUs map over the training images for which the ROI of the object is known thanks to the depth cues. A linear model is learn that finds the most likely position and size for the object by gradient descent starting at the global maximum of the VTUs map. The method is evaluated both

in synthetic imagery and in real online learning experiments. It is also compared to other methods in the UIUC cars (side view) database. It was found to perform slightly worse (approx. 3%) when compared to the methods of Leibe et al (2004) and Mutch and Lowe (2006). However it has to be taken into account that the Wersing et al approach use a codebook of 50 shape features instead of up to 1000 features as in Mutch and Lowe and Leibe et al. methods, making online learning and detection possible. The system has been demonstrated running in real time in an Honda Asimo robot.

2.4 Local Features

As we have seen in the previous section, the majority of recent successful object recognition algorithms make use of interest regions to select relevant and informative features of the objects to learn. In this section we will explain the most relevant feature detectors and descriptors.
Extracting and describing these features is a computationally demanding task, that can be prohibitive for applications that must run close to real-time. Fortunately, work is being devoted to achieve faster methods and to develop implementations of the algorithms ready to run in special hardware. For example in Heymann et al (2007), the authors designed a version of the SIFT feature detector and descriptor (Lowe, 2004) that can run at twenty 640x480 frames per second on a GPU unit. Even a FPGA that can be used to detect affine-covariant features at camera frame-rate (Cabani and MacLean, 2006) is being developed. These and further advances in these techniques will enable these features to be used in real-time applications.

2.4.1 Detectors

The Differences of Gaussians (DoG), proposed by Lowe (2004) together with the SIFT descriptor, is a similarity-invariant feature detector. In order to obtain scale invariance, points are detected at multiple scales in a scale-space constructed convolving the image with Gaussian kernels of different scale.
Then, features are found at the extrema of Difference of Gaussians function convolved with the image, $D(x, y, \sigma)$, computed by subtracting two nearby scales of the scale-space separated by a constant multiplicative factor k (see Figure 2.2):

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, k\sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma)$$

The Difference of Gaussians is known to provide a good approximation of the Laplacian of the Gaussian:

$$LoG(x, \sigma) = \sigma^2 \nabla^2 G$$

that has been shown to detect stable image features (Mikolajczyk, 2002). Candidate points are then adjusted to sub-pixel and sub-scale resolution by fitting a 3D quadratic funtion to determine the interpolated location of the maximum.

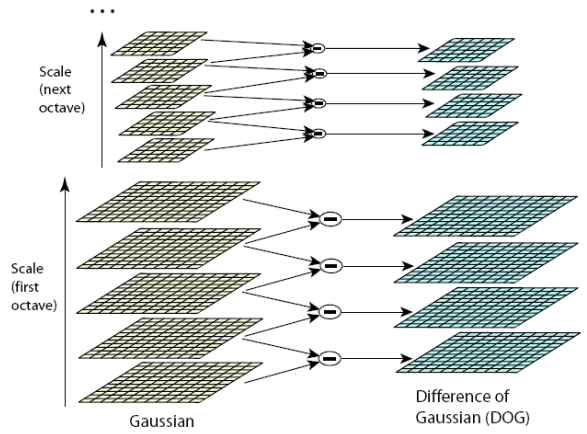


Figure 2.2: Schema of the Differences of Gaussians computation. At the left the initial image is incrementally convolved with Gaussians. The adjacent image scales are subtracted to produce the Differences of Gaussians, which are shown at the right. This figure has been taken from (Lowe, 2004).

Further computations for each feature are done at the Gaussian smoothed image with the closest scale, to ensure scale-invariance. Unstable extrema (i.e. with low contrast) and edge responses are rejected in order to increase robustness of the detected features to small amounts of noise. Next, *canonical* orientation of the detected feature is determined by clustering in a 36-bin histogram the orientation of the gradient of sample points within a region around the feature point.

The Harris Affine, proposed by Mikolajczyk and Schmid (2004) is a scale and affine covariant version of the corner detector proposed by Harris and Stephens (1988). A short explanation of the original algorithm follows.
The Harris Corner Detector is an improvement of the interest point detector by Moravec (1980) and locates the corners in the image. A corner can be described in terms of image intensities as a region where the intensity gradient changes fast in two nearly-orthogonal directions. The second moment matrix or auto-correlation matrix describes the distribution of the gradient in a local neighborhood. The two eigenvalues of this matrix (α and β in the equations) represent the magnitude of the two principal signal changes.

$$M = \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \tag{2.1}$$

Where I_x and I_y are the first derivatives of the image in the x and y directions averaged with a Gaussian window to prevent a noisy output. Depending on the eigenvalues of the matrix three situations may arise: both eigenvalues are low, which means that we are in a region of constant intensity (a flat region); one of

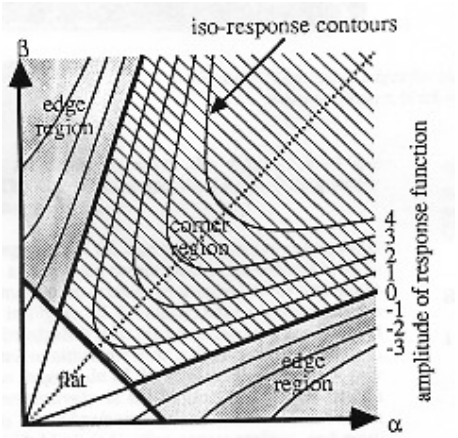


Figure 2.3: This figure, taken from the Harris and Stephens (1988) original article, shows the relation between the eigenvalues of the autocorrelation matrix (α and β) and the geometry of the image and the response of the Harris function.

the eigenvalues is high and the other low, this corresponds to an edge and, finally, when both eigenvalues are high we have a corner. The eigenvalue decomposition is, however, time-consuming. A faster way to compute the relation between the eigenvalues uses *Tr(M)* and *Det(M)*:

$$Tr(M) = \alpha + \beta \tag{2.2}$$

$$Det(M) = \alpha\beta \tag{2.3}$$

The corner response can then be formulated as:

$$R = Det(M) - kTr(M)^2 \tag{2.4}$$

where k is a constant parameter which tunes the response of the function usually put to 0.04. Figure 2.3 shows the response of this cornerness function for different values of α and β. An alternative way to compute the relation between the eigenvalues is

$$R = \frac{Det(M)}{Tr(M) + \epsilon} \tag{2.5}$$

where ϵ is a small value to avoid divisions by zero. This second method does not depend on any threshold. Applying this function to the whole image gives us the curvature of every pixel. Then, the corners can be detected as the local maxima of this response. Local features detected with this method demonstrated to be robust to changes in lightning and noise. However, as mentioned before, an affine covariant region must be defined around the point in order to reliably match corresponding corners.
The scale invariance Harris Laplace detector is obtained using the multi-scale approach proposed by Lindeberg and Gårding (1997). In this approach the

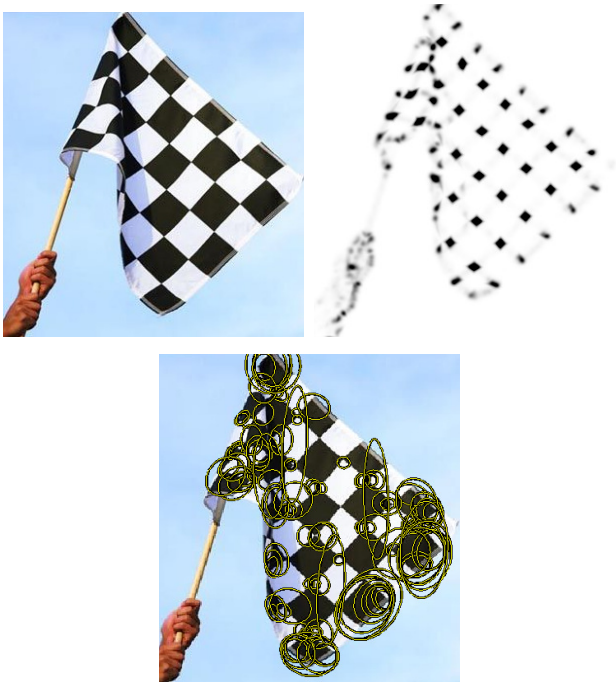


Figure 2.4: Example of the response of the Harris Affine. a) Original image. b) Response of the Harris corner detector (darker means higher response). c) Detected Harris-Affine regions.

Harris points are selected at their *characteristic scales* with the Laplacian of the Gaussian. The characteristic scale of a feature is the scale at which it is best represented. To select corners at their characteristic scales, first a scale-space representation of the Harris corner detector output must be constructed. That is why the second moment matrix is adapted to scale changes, to make it independent of the image resolution:

$$M = \mu(x, \sigma_I, \sigma_D) = \begin{bmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{bmatrix} = \sigma_D^2 g(\sigma_I) * \begin{bmatrix} I_x^2(x, \sigma_D) & I_x I_y(x, \sigma_D) \\ I_y I_x(x, \sigma_D) & I_y^2(x, \sigma_D) \end{bmatrix} \tag{2.6}$$

The local image derivatives are calculated with Gaussian kernels of scale σ_D (differentiation scale). The derivatives are then averaged by smoothing with a Gaussian of scale σ_I (integration scale). The scale at which corners are detected is determined by σ_D and, as in the Harris corner detector, the objective of the integration is reducing the sensitivity to noise. The scale-space is built with the Harris function for pre-selected scales $\sigma_n = \xi^n \sigma_0$ where ξ is a scale factor between successive levels, set to 1.4 by Lindeberg and Gårding (1997) and Lowe (1999) and to 1.2 in the computationally efficient version of the Harris Laplace

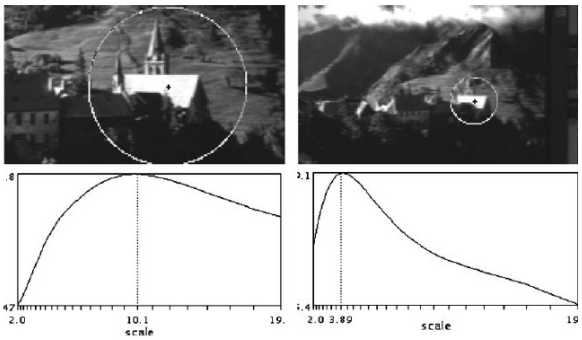


Figure 2.5: The response to the Laplacian of the Gaussian across scales for the feature detected in the images attains a maximum at its characteristic scale. This image has been taken from (Mikolajczyk et al, 2005)

algorithm (Mikolajczyk and Schmid, 2004). Then the matrix $\mu(x, \sigma_I, \sigma_D)$ is computed picking $\sigma_I = \sigma_n$ and $\sigma_D = s\sigma_n$ where s is a constant factor set to 0.7. Finally corners are detected at every scale by extracting the local maxima. Once the scale-space is constructed, the characteristic scale of each detected corner is automatically selected using the Laplacian-of-Gaussian:

$$|LoG(x, \sigma_n)| = \sigma_n^2 |I_{xx}(x, \sigma_n) + I_{yy}(x, \sigma_n)| \tag{2.7}$$

The Laplacian-of-Gaussian detects blob-like structures in the output of the Harris detector, when the size of the blob (produced by a corner) and the size of the LoG kernel over scales are equal, a maximum is attained. The characteristic scale is relatively independent of image resolution, it is related to the scale of the structure and not the resolution at which the structure is represented. The points where the LoG attains no extremum or the LoG response is below a threshold are rejected. Finally, Harris-Affine Detector aims to obtain the same set of pixels for every occurrence of a local structure. With this in mind, a procedure to undo the deformations introduced by changes in the point of view is needed. The solution proposed by Mikolajczyk and Schmid (2004) is based on estimating the *affine shape* of the local structure. Harris-Laplace automatically selects the scale of a feature, but can only deal with similarity transformations. However, when the scale changes in orthogonal directions are different (for example in a 3D rotation, when the object is slanted), it fails. At this point the shape of the region detector changes from a circle to an ellipse, to account for the different scaling in orthogonal directions. This effect can be appreciated in figure 2.6. To estimate the parameters of the ellipse that encloses the selected region, the second moment matrix is used. Lindeberg (1998) and Baumberg (2000) explored the properties of this matrix as an isotropy measure to find the affine deformation of an isotropic structure. Whitout loss of generality, a local anisotropic structure is assumed to be an affine transformed isotropic structure. Then, finding the transformation that projects the anisotropic pattern to an

Figure 2.6: In this image we can appreciate the different scaling in the orthogonal directions produced by 3D rotations (image taken from (Mikolajczyk et al, 2005)).

isotropic one, the parameters of the affine transformation can be recovered up to a rotation factor. This rotation can later be recovered using methods based on the gradient orientation (Lowe, 2004). The relation between the eigenvalues of the second moment matrix reveal the anisotropy of the region: if both eigenvalues are equal, the region is isotropic.
A really brief outline of the iterative algorithm is presented next. It must be taken only as a way to intuitively understand the behavior of the method, not as exact instructions for its implementation.

1. The *initial points* are selected using the scale-adapted Harris corner detector. These points are not detected in an affine invariant way, but its scale and location serve as initial values for further search.

2. At each iteration of the detection process, the *integration scale* of a corner is selected at the maximums of the normalized Laplacian of the Gaussian across scales.

3. The *differentiation scale* is selected at the maximum of normalized isotropy using the value of the integration scale σ_I and the relation between the eigenvalues of the second moment matrix. Different values of the differentiation scale σ_D are generated using $\sigma_D = s\sigma_I$ for values of s in the range $[0.5, .., 0.75]$. The value of s that produces the isotropy measure $Q = \frac{\lambda_{min}(\mu)}{\lambda_{max}(\mu)}$ closer to 1 is selected.

4. Once the scales are selected, the *spatial localization* of an interest point is refined in the affine-transformed domain and a displacement vector between the original point and the precise localization is back-projected to the original image domain. This vector is then used to correct the spatial localization of the point.

5. The *shape adaptation matrix* is estimated with the second moment matrix computed in the preceding steps. The transformations are computed as

$\mu(k) = \mu^{\frac{-1}{2}}(x^{(k)}, \sigma_I^{(k)}, \sigma_D)^{(k)}$ for step k of the iterative process. This transformation is then concatenated with the previous transformations using:

$$U = \Pi_k(\mu^{\frac{-1}{2}})^{(k)} U^{(0)} \tag{2.8}$$

The second moment matrix is determined by the values of σ_I and σ_D at each step, which are automatically selected at each iteration. Thus, the resulting μ matrix is independent of the initial values of scale and resolution of the image.

6. The *convergence criterion* can be based either on the U or the μ matrix. If it is set to μ, the algorithm stops when the relation between the eigenvalues is close enough to 1. In practice a small error of $\epsilon_C = 0.05$ is allowed:

$$1 - \frac{\lambda_{min}(\mu)}{\lambda_{max}(\mu)} < \epsilon_C \tag{2.9}$$

The second stopping criterion consists in decomposing $U = R^T DR$ and compare the consecutive U-transformations. If consecutive R and D transformations are similar enough, the iterative algorithm is stopped. Both termination criteria produce the same final results. In case of divergence a stopping criterion must be also set. If the eigenvalue ratio $\frac{\lambda_{max}(\mu)}{\lambda_{min}(\mu)} > \epsilon_C$ for $\epsilon_C = 6$, the point should be rejected as it leads to unstable elongated structures.

7. If termination criteria is not satisfied, go to step 2.

When the parameters of the ellipse are calculated, the affine covariant region is extracted and remapped to a circle. This circle is the normalized view from where the local descriptors will be computed.

The Hessian Affine covariant region detector (Mikolajczyk et al, 2005) works in the same way as the Harris Affine but, in this case, the feature regions detected are blobs and ridges instead of corners. The rest of the algorithm is the same as the Harris Affine: a scale-space selection to detect the characteristic scale of each point using the Laplacian of the Gaussian, and an elliptical affine region estimation using the relation between the eigenvalues of the second moment matrix. The difference between the Hessian Affine and the Harris Affine comes from the selected initial points. In this case the Hessian matrix is used:

$$H = H(x, \sigma_I, \sigma_D) = \begin{bmatrix} \mu_{11} & \mu_{12} \\ \mu_[21] & \mu_{22} \end{bmatrix} = \sigma_D^2 g(\sigma_I) * \begin{bmatrix} I_{xx}(x, \sigma_D) & I_{xy}(x, \sigma_D) \\ I_{xy}(x, \sigma_D) & I_[yy](x, \sigma_D) \end{bmatrix} \tag{2.10}$$

The second derivatives of the image give a strong response on blobs and ridges. The local maximums of the determinant of this matrix correspond to a blob-like structure. In addition, a function based on the determinant of the Hessian matrix penalizes very long structures for which the second derivative in a particular direction is very small. This type of structures are commonly unstable and difficult to locate precisely.

Figure 2.7: Some regions detected by the Hessian Affine covariant region detector.

MSER or the Maximally Stable Extremal Region detector proposed by Matas et al (2002) detects connected components at all the possible thresholding levels of an image. The concept of *extremal region* defines a set of pixels with a value either higher or lower than all the neighboring pixels, which can be seen as a local maximum or minimum (an extremum) of the surface defined by pixel intensities. Finally, *maximally stable* refers to extremal regions where the intensity values of the pixels of the region is several levels higher (or lower) compared to the neighbors. This type of local features has some desirable properties:

- Affine changes in illumination preserve the regions since they only depend on the ordering of the pixels and not on the intensity values.
- Geometric changes that can be locally approximated by an affine transformation, an homography or even a continuous non-linear warping will preserve the topology. This means that pixels from a single connected component will also be in a connected component in the transformed image.
- The *maximally stable* requisite ensures that noise or acquisition problems will not alter the regions significantly.
- Since no smoothing is involved in the process, both very fine and very large structures are selected.
- An efficient, near linear complexity, detection algorithm exists for this type of local feature. The complexity for detecting all the regions in an image is $O(n\ log\ log\ n)$ where n is the number of pixels in the image.

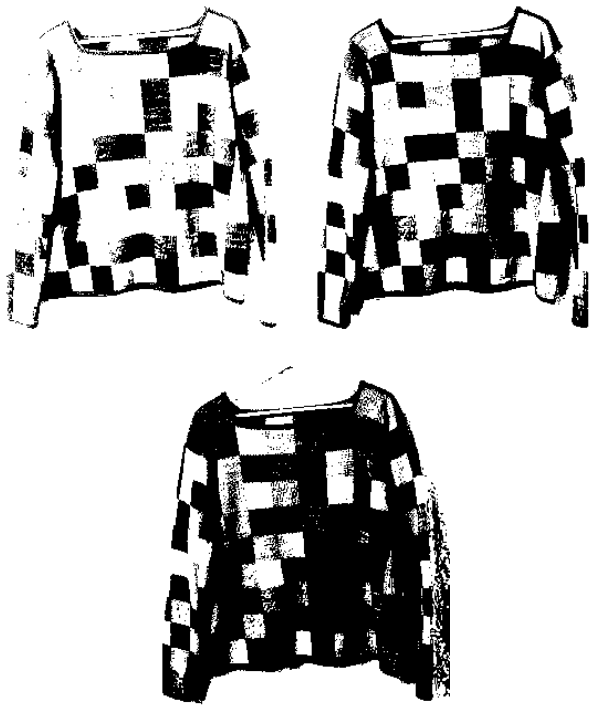


Figure 2.8: An image binarized at different threshold levels: a) 51 b) 77 and c) 102. The MSER regions are those that do not change (no pixel of its boundaries switches color) during several successive threshold levels.

The structure of the proposed algorithm is the following: First, all the pixels are sorted by its intensity value. Next, all pixels are marked in the image (either in increasing or decreasing order) and the *union-find* algorithm is used to maintain the growing and merging between the connected components as in the watershed algorithm. During the enumeration process, the area of each connected component as a function of the intensity is stored. Then, the maximally stable extremal regions are found by looking at the parts of these functions where no changes in the area of a connected component occur during a long range of intensity thresholds. This procedure is done twice, one for positive MSER regions (increasing order) and one for negative MSER regions (decreasing order).

Once each MSER region is detected, a measurement region is defined around it. These region can be of arbitrary size as long as it is constructed in an affine-invariant way. The purpose of this measurement regions is to define the area that will be used to construct the descriptors for the MSER regions. The size of the measurement region is a tradeoff between the risk of crossing a deep discontinuity or a non-planar region and distinctiveness. Smaller measurement

Figure 2.9: Some regions detected by the MSER affine covariant region detector.

regions are more likely to be planar, but in the same way they are much less likely to be unique or, at least, discriminative enough. To work out this problem, Matas et al (2002) propose the use of various measurement regions at different scales for every MSER region: The region itself, and the convex hull scaled 1.5, 2 and 3 times. The MSER regions are normalized to a circle using the second order moments of the ellipse that encloses the region.

The Speeded Up Robust Features (SURF) detector (Bay et al, 2008) is a hessian-based feature detector that, thanks to several optimizations like using an integral image to compute feature locations, can run at fast rates. Even though SURF includes both a detector and a descriptor, here we have only used the detector part, and evaluation of the SURF descriptor in our experiments is left as future work.

This detector finds similar regions to the Hessian Laplace, as is also based in the Hessian matrix. However, in the SURF approach, second derivatives of the Gaussian used to compute the Hessian matrix are replaced by box filters equal to Haar features (Haar features can be seen in Figure 2.10), that are straightforward to compute with an integral image as done by Viola and Jones (2001).

The approximated determinant of the Hessian matrix, computed with the box filters replacing the Gaussian second derivatives, is used to compute the response to blob-like structures at multiple scales. This structures will be later detected as the local maxima after a $3 \times 3 \times 3$ neighborhood non-maxima suppression.

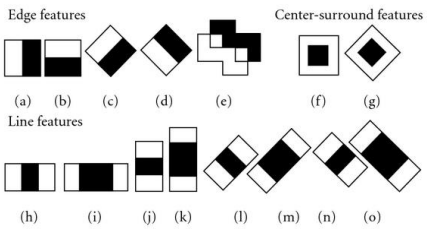


Figure 2.10: Haar features.

2.4.2 Descriptors

Local features are of little use if they can not be compared and matched with regions from other images. The vast majority of applications that employ local features such as wide baseline matching, object recognition, image retrieval, recognition of object categories or robot localization involve a matching step where features of two or more different images must be compared and put in correspondence. This step implicitly involves the use of a *local descriptor*. The objective of these descriptors is to provide a compact and distinctive representation of the local feature to simplify the matching stage and, at the same time, induce robustness to the remaining variations in the measurement regions. These variations can be illumination changes, noise and changes in the measurement region introduced by deep discontinuities or non-flat surfaces.
Abundant types of local descriptors are present in the literature with different degrees of complexity and robustness to error in the measurement region. The simplest possible descriptor is the region pixels alone, but this descriptor is very sensitive to noise and illumination changes. Other descriptors make use of histograms, image derivatives or information from the frequential domain to increase the robustness.
Recently, Mikolajczyk and Schmid (2005) published a performance evaluation of various local descriptors. In this review more than ten different descriptors are compared for affine transformations, rotation, scale changes, jpeg compression, illumination changes and blur. The conclusions of this analysis observe an advantage in performance of the Scale Invariant Feature Transform (SIFT) introduced by Lowe (2004) and its variant Gradient Location Orientation Histogram (GLOH) (Mikolajczyk and Schmid, 2005) and, at a certain distance, PCA-SIFT (Ke and Sukthankar, 2004). Follows a description of the SIFT algorithm. After the local feature detection, a set of pixels from the image (the measurement region) is extracted and normalized to a common representation. Then, from this normalized image patch, the descriptor is computed.

The SIFT descriptor is based on a model proposed by Edelman et al (1997) where biological vision is imitated. Complex neurons in the primary visual cortex are activated by a gradient in a particular orientation if it appears within a small

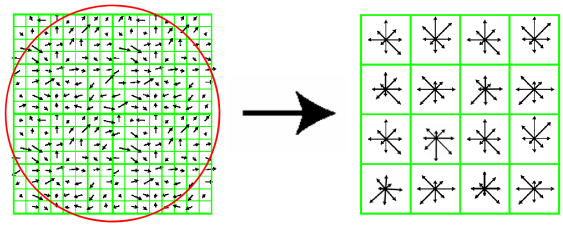


Figure 2.11: The SIFT algorithm divides the local patch in sub-regions with image gradients and histogram construction.

range of positions in the retina. The hypothesis presented by Edelamn et al is that this complex neurons function is to allow object recognition under small 3D rotations, which would introduce small displacements in the gradients position. Experiments performed using a computational model inspired by these ideas show an improvement in performance of more than 55% over direct correlation of the gradients.

Although based on the idea proposed by Edelman et al, the SIFT descriptor has a different computational model. In the algorithm proposed by Lowe, the image patch is divided into 16 sub-regions and an histogram of the orientations of the gradient is computed for every sub-region. A gradient can move within a sub-region and still produce the same descriptor, in this way the shift in position allowed by the complex neurons is emulated. The orientations are quantized by the magnitude of the gradient to lower the contribution of instable orientations from sample points in flat zones of the image. The histograms have eight bins, each of 45 degrees. In order to avoid sudden changes in the descriptor with small changes in the image position, and to give more importance to gradients on the center of the measurement region, a Gaussian weighting function with sigma one half the width of the measurement region is used to weigh the sample points. The reason to give more relevance to the central gradients is because these points are less likely to suffer registration errors introduced by non-planar surfaces and depth discontinuities. A trilinear interpolation is used to distribute gradient samples across adjacent bins of a histogram, to avoid boundary effects in the gradient orientation. In this way, a small change in the orientation will not change the descriptor abruptly.

Once all the histograms are constructed, the values of all bins are arranged as a vector. Sixteen histograms of eight bins each yield a vector of 128 dimensions. This vector is the descriptor used to identify every local feature. To achieve illumination invariance, the vector is normalized to unit length, this normalization will make the descriptor invariant to affine changes in the intensity of the measurement region. However, non-affine illumination changes such as camera saturation or different reflectance from 3D surfaces with different orientations can cause changes in the magnitude of some gradients, but it will rarely affect the orientation of the gradient. To reduce the influence of these variations, each

value of the normalized feature vector is thresholded to 0.2. The value of 0.2 was determined experimentally using images containing 3D objects under different illumination conditions.

Gradient Location Orientation Histogram (GLOH) proposed by Mikolajczyk and Schmid (2005) is an extension of the SIFT descriptor. The algorithm to compute the descriptor is the same except for the distribution of the sub-regions. Here a log-polar location grid is used, with three sub-regions in radial direction, each divided into 8 sub-regions in angular direction except the central sub-region. This results in 17 sub-regions. Instead of 8 orientation bins for each histogram, 16 bins are used. The resulting feature vector has 272 values, which are reduced to 128 with PCA.

Chapter 3

Global Localization Method

In this chapter we propose a topological vision-based localization approach for a mobile robot evolving in dynamic indoor environments. Robot visual localization and place recognition are not easy tasks, and this is mainly due to the perceptive ambiguity of acquired data and the sensibility to noise and illumination variations of real world environments. We propose to approach this problem by using a combination of affine covariant detectors so as to extract a robust spatial signature of the environment.

The global localization system proposed is similar in spirit to the one proposed by Tapus et al (2004); Tapus and Siegwart (2006). In their work a *fingerprint* is defined as a generic descriptor of a room. This descriptor is a circular string, composed by a character coding each occurrence of a determined feature. They propose color blobs and vertical lines, extracted from an omnidirectional image, and edges, extracted from a laser range-scanner reading, as features. The advantage of our method is that we only use an omnidirectional vision sensor, and therefore the cost of expensive sensors (two laser range-scanners) is eliminated. In addition, the information extracted from the images to characterize the rooms can be used by other high-level processes such as object recognition, thus reducing the computational load of the robot.

The proposed representation to characterize a place is a *constellation* of feature regions extracted from a panoramic image of the room. The local nature of this representation makes it robust against partial changes in the image due to occlusions, change in point of view or dynamic changes in the environment. We decided to use combinations of the following three feature region detectors: MSER (Maximally Stable Extremal Regions) (Matas et al, 2002), Harris-Affine (Lindeberg, 1998), and Hessian-Affine (Mikolajczyk and Schmid, 2004), that have been explained in Section 2.4. These region detectors have shown to perform better when compared to others.

When a new signature is acquired, it is compared to the stored panoramas from the *a priori* map. The panorama with the highest number of matches is selected. To improve the results and discard false matches, the essential matrix is computed and used to filter the outliers. Finally, the panorama with the

highest number of inliers is selected as the best match.
In our approach images are acquired using a rotating conventional perspective camera. When a set of images covering 360 degrees is acquired, they are projected to cylindrical coordinates and the feature regions are extracted and described. The descriptors constellation is next constructed automatically. Hence, by using feature regions to construct the signature of a location, our approach is much more robust to occlusions and partial changes in the image than the approaches using global descriptors. This robustness is obtained because many individual regions are used for every signature of a location and, thus, if some of them disappear the constellation can still be recognized.
Nevertheless, combining different region detectors increases the computational time and memory requirements. For this reason we show that a re-ranking mechanism based on a global appearance-based similarity measure can be used to prioritize the most similar map nodes. This approach is compatible with techniques such as the incremental spectral clustering, proposed by Valgren and Lilienthal (2008) to reduce the number of stored panoramas and construct a topological map from the raw visual data.
This framework gives us an interesting solution to the perceptual aliasing problem (one of the main difficulties when dealing with qualitative navigation and localization). Our approach is validated in real world experiments and is compared to other vision-based localization methods.

3.1 Panoramic Image Acquisition

Instead of using an omnidirectional camera, the panoramas have been constructed by stitching together multiple views taken from a Sony DFW-VL500 camera mounted on a Directed Perception PTU-46-70 pan-tilt unit. The camera and pan-tilt unit can be seen in Figure 1.1.
In order to build a panorama using a rotating camera, it had to be taken into consideration that the image sequence employed must have a fixed optical center. Translations of the optical center would introduce motion parallax, making the image sequence inconsistent. However, if the objects in the scene are sufficiently far from the camera, small translations can be tolerated. The steps to stitch all the images in a panorama are the following:

1. The first step consists of projecting all the images of the sequence to a cylindrical surface. The points are mapped using the transformation from Cartesian to cylindrical coordinates:

$$\theta = tan^{-1}(\frac{x}{f}), \qquad v = \frac{y}{\sqrt{x^2 + f^2}} \tag{3.1}$$

where x and y are the position of the pixel, f is the focal distance measured in pixels and θ and v are respectively the angular position and the height of the point in the cylinder. The cylinder radius is the focal length of the camera used to acquire the images, as in this way the aspect ratio of the

image is optimized (Shum and Szeliski, 1997). Taking this into account, the size of the panoramas acquired by our system have a size of 5058x500 pixels.

2. Once all the images have been projected to cylindrical coordinates, the rotation between each pair of images must be estimated. In principle, only panning angles need to be recovered but, in practice, to correct vertical misalignment and camera twist, small vertical translations are allowed. Therefore, a displacement vector $\Delta t = (t_x, t_y)$ is estimated for every pair of input images. The implemented method to compute Δt distinguishes between three situations:

   i If sufficient feature points are found in the shared part of the images, Δt is computed by means of matches between pairs of feature points. To find the translation with most support among matches, and to exclude false matches and outliers, RANSAC is used.

   ii In those cases where there is not enough texture in the images to extract sufficient feature points, Δt is computed looking for a peak in the normalized correlation between the edges detected by the Canny edge detector (Canny, 1986) of the two images. This method has the advantage over other correlation-based approaches of being independent of the illumination conditions and the vignetting effect (intensity decreases towards the edge of the image). In addition, as all the image is used, even with small amounts of texture a reliable translation can be estimated. However, this technique is computationally more expensive than feature matching and is not invariant to rotations or other deformations in the image.

   iii If no texture exists at all and the above procedure fails, the only remaining solution is to compute the expected translation if the angular displacement φ (in radians) between the images is known: $t_x = f\varphi$ and $t_y = 0$

3. Due to automatic camera gain, vignetting or radial distortion, an intensity jump may appear between two images as can be seen in Figure 3.1. In this work the most straightforward solution is taken, that consists in blending linearly every two consecutive images. This method produces results good enough for visualization purposes and is suitable for static scenes. However techniques such as multi-band blending and deghosting as the ones proposed by Shum and Szeliski (1997), Brown and Lowe (2003), Szeliski and Shum (1997) or Uyttendaele et al (2001) can be used to improve the result by eliminating stitching artifacts and dynamic objects that created *ghosts* in the panorama.

Although the panoramic images were constructed for validation purposes, the constellations of feature region descriptors were not extracted from them. Instead, the features from the original images projected to cylindrical coordinates

Figure 3.1: Intensity jumps between successive images caused by automatic camera gain. Applying linear blending solves the problem.

were used. The reason for this is to avoid false regions introduced by possible new artifacts created during the stitching process. The panoramas built with the stitching method were all correctly constructed, even in the case of changes in lightning, reflections, multiple instances of objects or lack of texture.

3.2 Panorama Matching

The region detectors and descriptors provided by Mikolajczyk and Schmid (2005)[1] were used to extract the affine-covariant regions from the images and compute the SIFT descriptor vectors.
The procedure to compare two panoramas is relatively straightforward. First, matches are established as nearest neighbors between the feature descriptors of both panoramas using the Euclidean distance as similarity measure. Potentially false matches are rejected comparing the distance of the first and the second nearest neighbor in the same way as proposed by Lowe (2004). Additionally, reciprocal matching is used to filter even more false matches: if feature f_a from the first panorama matches feature f_b of the second panorama, but feature f_b does not match feature f_a, the match is discarded.
Next, the epipolar constraint between the panoramas is enforced by computing the essential matrix. The most straightforward way to automatically compute the essential matrix is using the normalized 8-point algorithm (Hartley and Zisserman, 2004). However, assuming that the robot will only move through flat surfaces, it is possible to use a simplified version where only 4 correspondences are necessary.

$$E = \begin{bmatrix} 0 & e_{12} & 0 \\ e_{21} & 0 & e_{23} \\ 0 & e_{32} & 0 \end{bmatrix} \tag{3.2}$$

[1] `http://www.robots.ox.ac.uk/~vgg/research/affine/`

Therefore, with a set of at least four correspondences of points of the form

$$p = [x, y, z] = [sin(2\pi\tilde{x}), \tilde{y}, cos(2\pi\tilde{x})] \tag{3.3}$$

where $\tilde{x}$ and $\tilde{y}$ are the normalized point coordinates in the planar panorama image, the following equations can be written:

$$\begin{bmatrix} y'_1x_1 & x'_1y_1 & z'_1y_1 & y'_1z_1 \\ \vdots & \vdots & \vdots & \vdots \\ y'_nx_n & x'_ny_n & z'_ny_n & y'_nz_n \end{bmatrix} \begin{bmatrix} e_{12} \\ e_{21} \\ e_{23} \\ e_{32} \end{bmatrix} = 0 \tag{3.4}$$

where (x_i, y_i, z_i) and (x'_i, y'_i, z'_i) is the i^{th} pair of corresponding points. As outliers may still be present among the matches, RANSAC is used to automatically compute the essential matrix with most support. Finally, the set of inlier feature matches that agree with the epipolar constraint is used as the evidence of the relation between the two panoramas.

Given the high dimensionality of the feature descriptors, matching is expensive in terms of computational cost even for a small set of nodes. An alternative to exhaustive matching is to use a global similarity measure to re-rank the map nodes and estimate the essential matrix only for the k top map nodes or, applying an *any-time* algorithm approach, until a node with a certain ratio of inliers is met. The global similarity measure should be fast to compute and exploit the differences between the map nodes to improve the re-ranking. We have applied the Vocabulary Tree proposed in Nister and Stewenius (2006) for object categorization to re-rank the map nodes for a new query image as it fulfilled both requirements. In short, this method constructs a visual vocabulary tree of feature descriptors applying hierarchical k-means on a training dataset. Next, images are described as a normalized histogram of *visual word* counts. To give more emphasis to discriminative *visual words*, they are weighted using a Term Frequency-Inverse Document Frequency (TF-IDF) approach. This method is also used for object recognition in this work, and therefore is explained in greater detail in Chapter 6. Finally, the images in the training set can be re-ranked according to its Euclidean distance to the new image visual word histogram.

Although the presented method has a very good performance in our experiments, it is time-consuming to acquire a panorama rotating a pan-tilt unit every time a localization has to be performed. Instead, we evaluated the decrease in performance using uniquely a normal planar perspective image of 45° field of view to localize the robot.

The simplest way to decide the corresponding node is by the maximum number of matches after computing the essential matrix (Ramisa, 2006; Ramisa et al, 2008a; Valgren and Lilienthal, 2008). An alternative we tried was to use the ratio between the number of matches and the lowest number of keypoints of the two images (Booij et al, 2007). Experimentally, we did not find much difference between both approaches in our dataset and therefore we have retained the first one.

3.3 Experimental Design

The objective of the present chapter is twofold: On the one hand, we want to validate the proposed method for indoor global localization and, on the other hand, we target to experimentally determine if using different region detectors simultaneously improves significantly the localization results. Although successive images acquired by the robot while moving in the room could be used to incrementally refine the localization, in our experiments, we wanted to evaluate if combining different region detectors improves the robustness to viewpoint change for the presented global localization method and, therefore, we have only considered the worst case scenario, where only one image per room is available to localize the robot.

3.3.1 Dataset description

The test-bed data used in this work consists of 17 sequences of panoramas from rooms in various buildings[2]. Each sequence consists of several panoramas acquired every 20 cm following a straight line predefined path. This type of sequences are useful to check the maximum distance at which a correct localization can be performed. The sequences have been acquired in uncontrolled environments. In order to make the data set as general as possible, rooms with a wide range of characteristics have been selected (e.g., some sequences correspond to long and narrow corridors, while others have been taken in big hallways, large laboratories with repetitive patterns and others in smaller rooms such as individual offices). Panoramic images of the environment are shown in Figure 3.2. A short description of each sequence is given below:

- **iiia01** consists of 11 panoramas, and the sequence has been taken in a large robotics laboratory type of space.
- **iiia02** and **iiia03** contain 14 panoramas each, and have been taken at the conference room of the IIIA. In our experiments only the map node of iiia02 is used.
- **iiia04** is 19 panoramas long, and has been acquired in a long and narrow corridor.
- **iiia05** and **iiia06** have 25 and 21 panoramas, respectively. They have been taken in the library of the IIIA, the first one is from the library entrance and librarian desk, while the second is from a narrow corridor with book shelves. Both share the first panorama of iiia05 as map node.
- **iiia07** is 19 panoramas long. This represents another section of the robotics laboratory, and corresponds to a small cubicle.
- **iiia08** is 10 panoramas long, and has been acquired in a small machinery room.

[2]The data-set can be downloaded from `http://www.iiia.csic.es/~aramisa`.

- **iiia09** has 21 panoramas that have been taken at the back entrance hall. This sequence has been taken in a tilted floor, which is a challange for the 4-point algorithm, because of the flat world assumption.
- **iiia10** is 19 panoramas long and has been taken in the coffee room.
- **iiia11** has 21 panoramas and has been acquired in the entrance hall of the IIIA.
- **cvc01** is 21 panoramas long and corresponds to a long corridor of the CVC research center. As one of the corridor walls is made out of glass, the view field is wider than a normal corridor. However, direct sunlight affects the white balance of the images.
- **cvc02** is 21 panoramas long, and has been acquired in a large office with many desks.
- **cvc03** has 14 panoramas taken in a small office with just one working desk.
- **cvc04** has 22 panoramas and has been taken in a wide corridor with posters.
- **etse01** is the main hall of the engineering building and is 20 panoramas long.
- **etse02** has 21 panoramas and has been taken in a very wide corridor of the engineering building.

3.4 Results

In order to test the proposed method, we evaluated all possible combinations of the three selected region detectors with two different descriptors.
Table 3.1, shows the average percentage of correctly classified test panoramas for each combination. Results are provided using the 8-point algorithm, the 4-point algorithm and also the later with reciprocal matches. From the results illustrated in the table, it can be seen that by reducing the number of false matches with the reciprocal matches technique, improves substantially the performance. Therefore from now on, we only show the results obtained with this technique.
Standard deviation is also provided in order to assess the stability of combinations along the different sequences. Not much difference is observed among the descriptors GLOH and SIFT, which performed similarly in all cases. Looking at the feature detectors individually, the best results have been obtained by Harris Affine, while Hessian Affine and MSER had a similar performance. Overall, the

Combination	8 points algorithm		4 points algorithm		4 points and recipr. match	
	acl	std	acl	std	acl	std
HA+S	74%	23%	69%	23%	82%	22%
HA+G	70%	21%	73%	24%	81%	21%
HE+S	58%	24%	73%	26%	75%	25%
HE+G	63%	26%	65%	27%	74%	26%
M+S	62%	28%	78%	18%	76%	23%
M+G	61%	29%	69%	23%	74%	26%
HA+HE+S	64%	15%	78%	19%	86%	14%
HA+HE+G	67%	14%	79%	21%	87%	16%
M+HE+S	56%	23%	75%	23%	87%	15%
M+HE+G	60%	23%	78%	18%	88%	14%
M+HA+S	65%	21%	79%	19%	86%	14%
M+HA+G	70%	25%	79%	19%	88%	11%
M+HA+HE+S	62%	16%	82%	19%	89%	11%
M+HA+HE+G	64%	20%	82%	19%	90%	11%

Table 3.1: Average percentage of correctly localized panoramas (acl) across all sequences and standard deviation (std). For convenience we have labeled M: MSER, HA: Harris-Affine, HE: Hessian-Affine, S: SIFT, G: GLOH.

combinations of detectors outperformed the individual detectors. The best performance in the localization test has been achieved by the combination of the three detectors, which classified correctly 90% of the panoramas. This performance is mainly due to their good complementarity. Furthermore, in Figure 3.3 the average performance of two selected combinations is compared to the standalone detectors as a function of the distance to the map node. As can be seen, combinations cope better with changes in point of view than individual detectors. Sequences acquired in large rooms typically achieved a good performance

no matter the combination used. However, small rooms and specially long and narrow corridors seem to be more difficult environments, even if they are well textured. This can be explained because the distance between the robot and the perceived objects is short and, therefore, the objects' appearance changes rapidly resulting in an unreliable matching in the lateral regions of the panorama.
Some particularly difficult sequences have been cvc01, iiia04, iiia06 and iiia09. Table 3.2 shows the results with these sequences. As we can see, the performance is notably increased by combining different detectors. On average, standalone detectors achieved around 55%, while combinations increased to around 81% in these environments. The sequence iiia04 is a specially long and narrow corridor with very few texture; it is interesting to notice that in this case the combination of all feature types achieved 100% correct classification. Another notable finding is the extremely good performance of MSER on cvc01 when compared to the other detectors. Most of the similar approaches to global localization,

Combination	cvc01	iiia04	iiia06	iiia09
HA+S	35%	42%	75%	75%
HA+G	35%	47%	70%	70%
HE+S	20%	47%	75%	45%
HE+G	20%	53%	80%	30%
M+S	85%	42%	30%	65%
M+G	95%	53%	35%	35%
HA+HE+S	80%	84%	70%	95%
HA+HE+G	45%	89%	75%	85%
M+HE+S	90%	84%	80%	65%
M+HE+G	90%	84%	90%	60%
M+HA+S	90%	79%	70%	85%
M+HA+G	90%	89%	65%	80%
M+HA+HE+S	80%	95%	75%	80%
M+HA+HE+G	75%	100%	90%	75%

Table 3.2: Average percentage of correctly localized panoramas for some interesting sequences. The naming convention is the same as in Table 3.1.

like (Booij et al, 2007), use feature detectors only invariant to scale but not affine covariant, mainly because of its more expensive computational cost. For comparability, we have evaluated the performance of the Difference of Gaussians detector with the SIFT descriptor (Lowe, 2004). For our tests we used the implementation provided by Lowe[3]. It must be noted that the SIFT descriptor of this implementation is more robust than the one used for the remaining feature detectors used in this chapter, as can be seen in Figure 5.3 in Chapter 5. The performance of the DoG detector with the same descriptor used for the other feature detectors should be tested in future work. On average, using points detected with the DoG and SIFT, the correct location was selected in 72% of the

[3] `http://www.cs.ubc.ca/~lowe/keypoints/`

cases. However, it had an irregular performance depending on the environment type (27% standard deviation), with perfect results in large rooms, but very poor results in narrow corridors and small rooms. This was an expected outcome as this detector is less resistant to viewpoint changes.
In terms of computational complexity, the most expensive step of the approach is clearly the bidirectional descriptor matching as can be seen in Table 3.4. These computational times have been obtained with a C++ implementation of the method running in a Linux Pentium 4 at 3.0 GHz computer with 2Gb of RAM.

Combination	Matching (seconds)	RANSAC (milliseconds)
HA+S	4,31	3,046
HA+G	4,29	2,597
HE+S	2,87	3,016
HE+G	2,88	2,631
M+S	1,24	2,920
M+G	1,24	2,310
HA+HE+S	7,16	6,625
HA+HE+G	7,16	5,401
M+HE+S	4,11	5,827
M+HE+G	4,11	5,361
M+HA+S	5,51	6,682
M+HA+G	5,51	5,382
M+HA+HE+S	8,44	1,3941
M+HA+HE+G	8,47	1,0815

Table 3.3: Average feature matching and RANSAC time for each map node. It is important to notice the difference in time scale.

3.4.1 Re-ranking of map nodes

As explained in Section 3.2, the global appearance based image similarity measure from Nister and Stewenius has been used to re-rank the map nodes and prioritize those that appear more similar. We have build the vocabulary tree with Harris Affine features.
When used for object classification, this type of approach requires at least tens of training images in order to correctly determine the class of a novel object instance. However, we only used the map nodes to train both the vocabulary tree and the classifier. This gives only one training instance for each class. Despite so limited training data, the approach achieved the notable overall result of re-ranking the correct node in the first position for 62% of the query panoramas, and among the top five nodes 85% of times as can be seen in Figure 3.4. More detailed results of this re-ranking experiment are in Figure 3.6, where the performance is shown for each individual sequence.

As expected, the percentage of times the correct map node is re-ranked at the top position decreases as distance to the query panorama increases (see Figure 3.5).

3.4.2 Localization with 45° FOV images

Constructing a panoramic image with a rotating camera on a pan-tilt unit is a time-consuming step that requires the robot to stay in a fixed position during the acquisition. In order to assess the decrease in performance that would cause using just a single conventional image to localize the robot we have done the following experiment: For every test panorama, a random area that spans 45° and has at least 100 features is extracted and matched to the map nodes. This procedure is repeated for every test panorama. After a 10 repetitions experiment with all test panoramas, the average number of correct localizations was 73% using Harris Affine combined with MSER and the GLOH descriptor. This result is good considering how limited the field of view is. In addition to the time saved in image acquisition, the matching time is reduced almost one order of magnitude on average.

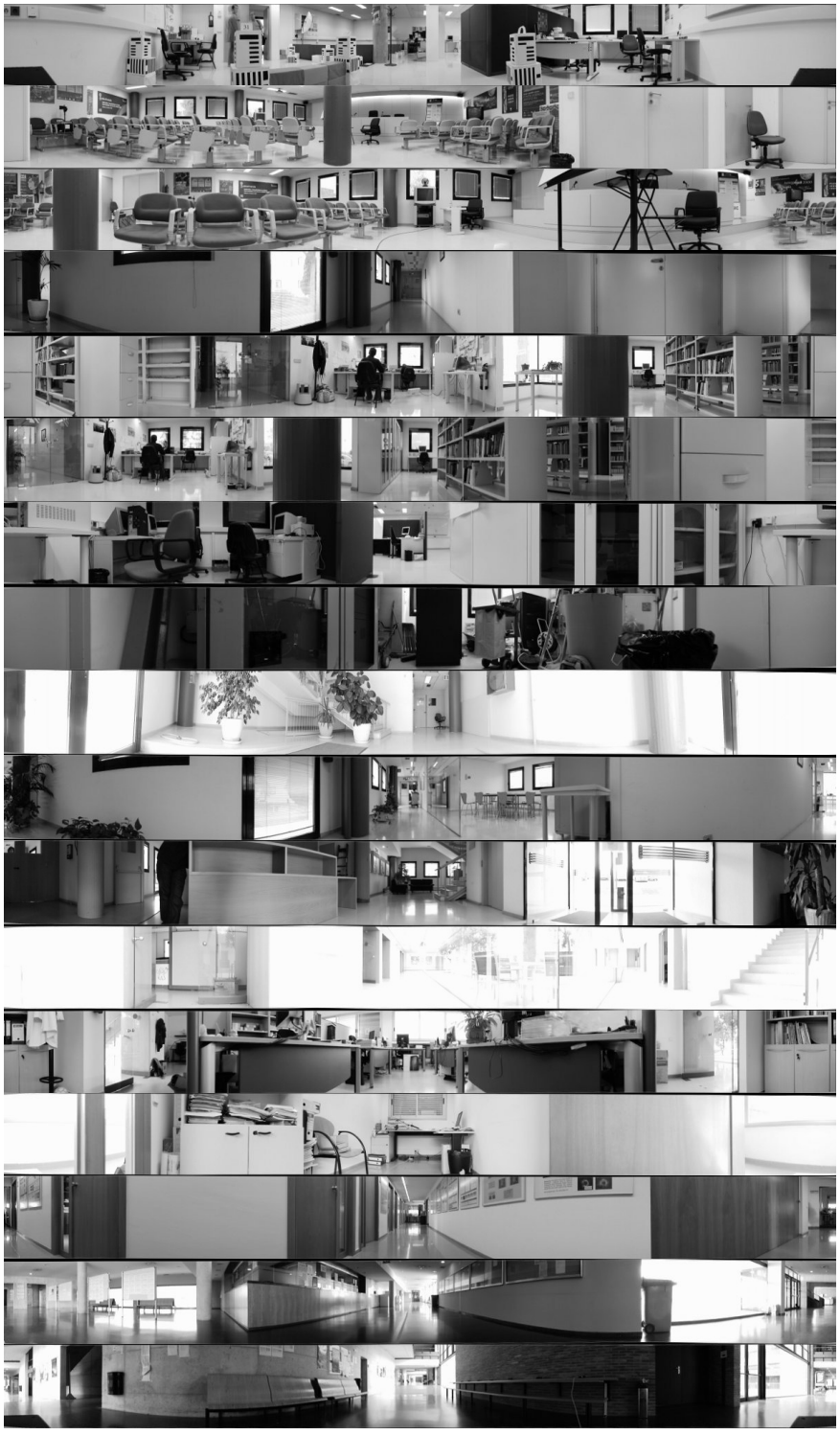


Figure 3.2: Panorama nodes in the same order as described in the text.

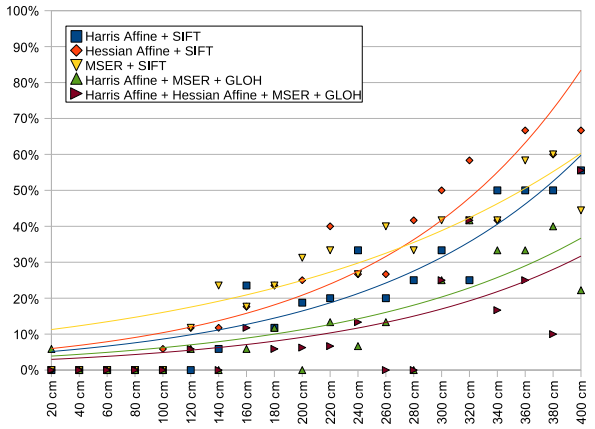


Figure 3.3: Percentage of incorrectly classified test panoramas as a function of the distance to the map node. The exponential regression of the data points is also provided for clarity.

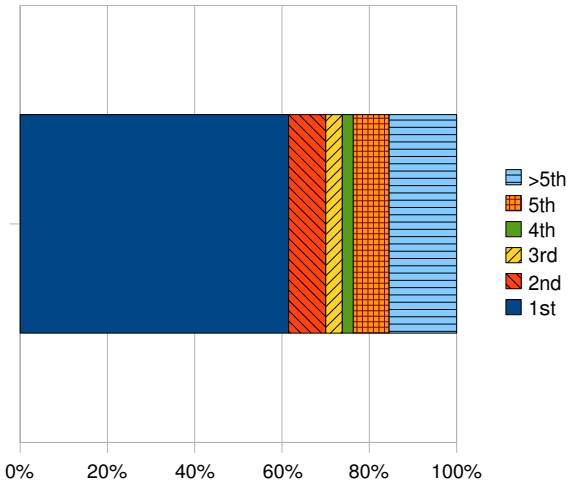


Figure 3.4: Position of the correct map node after re-ranking using the vocabulary tree.

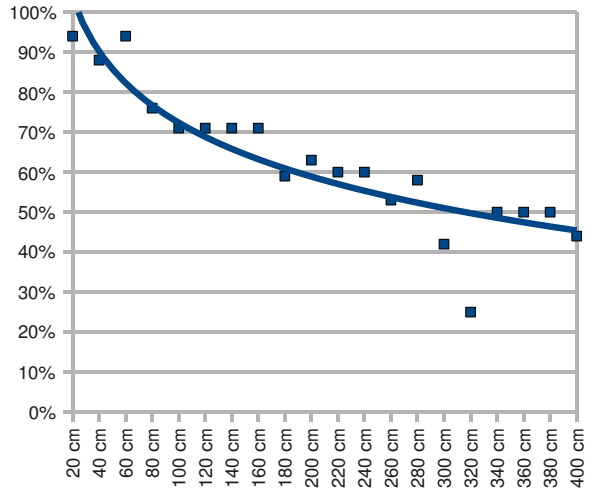


Figure 3.5: Ratio of query images with the correct node re-ranked at the top position against distance to first panorama of the sequence. The logarithmic regression curve is also shown.

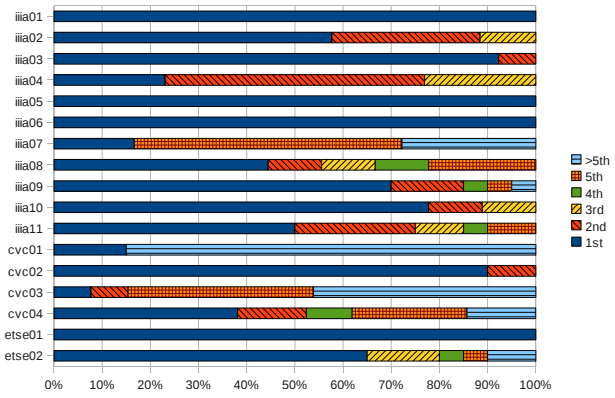


Figure 3.6: Position of the correct map node after re-ranking using the vocabulary tree per sequence.

Chapter 4

Appearance-Based Homing with the Average Landmark Vector

In order to navigate from one to the next node of the ones proposed in Chapter 3, one could factorize the essential matrix as done in works such as (Booij et al, 2006) to obtain the direction of movement to the next node. However this forces the robot to compute the essential matrix at every step, even if it is already correctly localized.

To complement the global localization method proposed in this work, we have investigated a biologically inspired homing method, the *Average Landmark Vector* (ALV), that can be used to travel between the nodes of the map graph. Lambrinos et al (1998, 2000) suggest the Average Landmark Vector as a way to model the navigation techniques of bees. This model assumes that the animal stores an average landmark vector instead of a snapshot image, as previous models by Carwright and Collet (1983) suggested. The advantages of this model are its simplicity, that only the orientation and the ALV at the home location have to be stored instead of a whole image. A third advantage is that no matching of the landmarks has to be done.

In robot homing research artificial landmarks are often used. This is a strong limitation as it requires setting up the environment beforehand. Instead, in this work the goal is to create a simple homing method that can be used without having to rely on artificial landmarks. For this we propose the combination of the ALV homing technique with visual invariant feature detectors, like the ones described by (Mikolajczyk et al, 2005), in panoramic images.

Experiments with the ALV homing method were first done in simulation (Goldhoorn et al, 2007a,b) and because the results were promising, experiments were also done with real robots (Goldhoorn, 2008) in an office environment. Additionally, experiments with artificial landmarks were also done for comparison purposes.

4.1 Average Landmark Vector (ALV)

In this section we describe the biologically inspired homing technique Average Landmark Vector by Lambrinos et al (1998, 2000). The ALV is defined as the average of the landmark (or feature) position vectors:

$$ALV(F, \overrightarrow{x}) = \frac{1}{n} \sum_{i=0}^{n} \overrightarrow{f_i} \tag{4.1}$$

Where $F = \{\overrightarrow{f_1}, \overrightarrow{f_2}, \ldots, \overrightarrow{f_n}\}$ is the collection of features that define the signature taken at the current position $\overrightarrow{x}$ and f_i are the coordinates of the i^{th} landmark position vector. In this equation F contains the global feature positions to explain and proof the homing technique. This is the robot centred version, but it is made world centred by subtracting the current position $\overrightarrow{x}$ to easily proof that the homing technique works :

$$ALV(F, \overrightarrow{x}) = \frac{1}{n} \sum_{i=0}^{i=n} \overrightarrow{f_i} - \overrightarrow{x} \tag{4.2}$$

To differentiate between the world coordinate system and the (self centred) coordinate system of the robot, the home vector is defined as follows:

$$homing(F, \overrightarrow{x}, \overrightarrow{d}) = ALV(F, \overrightarrow{x}) - ALV(F, \overrightarrow{d}) \tag{4.3}$$

Where $\overrightarrow{x}$ is the current location of the robot and $\overrightarrow{d}$ the destination. When the ALV functions are substituted by Equation 4.2 then $\overrightarrow{d} - \overrightarrow{x}$ remains, which is exactly the home vector. Figure 4.1 shows an example of the calculation of the home vector. To simplify, the image only the average landmark (the gray square) is shown. In this example it is also assumed that the depth of the landmarks is known. The ALVs are calculated for the current (C) and the $Home$ position, these are A_1 and A_2 respectively. The home vector (H) is calculated by subtracting the ALV at the destination position (A_2) from the ALV at the current position (A_1). This results in the home vector H which points to the destination location.

One important prerequisite of the ALV is that it is necessary to have the panoramic images aligned to an external compass reference before computing the homing direction. The Sahara ant *Cataglyphis*, for example, uses the polarization patterns of the blue sky to obtain the compass direction (Wehner, 1994).

ALV homing does not work when the ALV at the current location and at the goal location are the same (after correction for orientation differences), because this results in a zero vector. An exceptional theoretical case in which this could happen is when the ALV point, the current location and the goal location are aligned, in practice however this is very unlikely. To let the robot move anyway in such situations a random vector could be used to move the robot a small distance, and then continue the homing procedure.

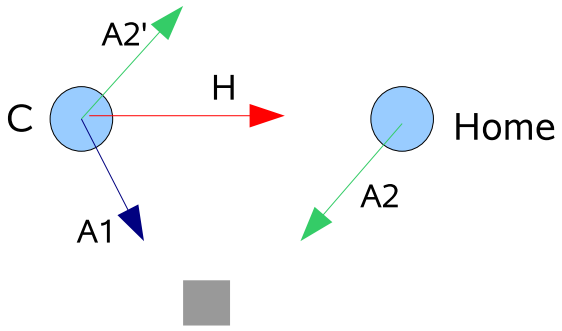


Figure 4.1: The calculation of the home vector. Both ALVs (A_1 and A_2) point to the average feature position, which is drawn as a gray block. The home vector (H) is calculated by subtracting the ALV at the destination location (A_2) from the ALV at the current location (A_1). This subtraction is shown, by the addition of the reverse vector, A_2', to A_1. The robots are aligned in this example.

In this work we propose to use the ALV method with natural feature points automatically extracted form images acquired with the mobile robot camera, without the need of artificial landmarks in the environment.
The feature points evaluated are the Differences of Gaussians from Lowe (2004) and the Maximally Stable Extremal Regions from Matas et al (2002). Only the x and z coordinates of the feature points are used to compute the ALV because of the flat world assumption. These local feature points possess qualities which make them interesting for the ALV. In the first place they are fast to compute (and even faster hardware-based approaches are being built), the second is that many higher-level processes are based on information from these interesting regions. Examples could be global localization (Ramisa et al, 2008a; Valgren and Lilienthal, 2008) or object recognition (Lowe, 2004; Csurka et al, 2004). Therefore there is no overhead in reusing them for the ALV. As a way to solve the constant orientation prerequisite, in our work all test panoramic images have been acquired with the robot facing a constant direction as is common practice in similar works (Möller et al, 2001; Hafner and Moller, 2001). In order to apply the ALV method in a navigation experiment a magnetic compass, or another system to acquire the global orientation, is required to align the panoramas.

4.2 Related Work

To the best of our knowledge no other work has addressed the combination of the ALV homing method with invariant feature points such as the MSER or the DoG.
So far, in most works that studied the ALV homing method, artificial landmarks have been used. For example Lambrinos et al (2000) used as landmarks four

black vertical cylinders, and in (Möller, 2000) experiments were done inside of a white box with several wide black vertical stripes on the walls. Möller et al (2001) did extensive experiments in a desert type outdoor scenario with four black cylinders as landmarks. In this same work an experiment was attempted in an indoor scenario. Natural landmarks were found by vertically averaging a certain area of the image and finding edges (i.e. intensity jumps) in the unidimensional graylevel profile.

Hafner and Moller (2001) investigated if a Multi-Layer Perceptron with back-propagation and a Perceptron with Delta Rule were able to learn a homing strategy both in simulation and in real world experiments. For the real-world experiments panoramic images acquired by the robot camera were reduced to a single line by vertically averaging (similarly to what Möller et al (2001) did), thus the input of the neural networks is a unidimensional image. Both neural networks successfully learned a homing strategy with the same characteristics as ALV.

Usher et al (2003) used a version of ALV augmented with depth information to guide a car-like vehicle in an outdoor experiment. Landmarks were salient color blobs and the depth information was acquired directly from the distance of the landmark to the center of the omnidirectional image (no unwrapping is performed) using a flat-world assumption. The authors performed real-world experiments using red traffic cones (witch hat model) as landmarks.

Vardy (2005) did an extensive study for a variety of biologically plausible visual homing methods in his PhD thesis, both for local and associative methods, in a real office environment. Among the methods evaluated in his work, there is the one proposed in Hafner and Moller (2001), referred to as *Center of Mass ALV*. In the experiments it performed similarly to other local homing methods, although it was found that an extra learning phase was necessary to determine which area of the panoramic image should be used to generate the unidimensional image in certain environments.

4.3 Experiments Performed and Results Obtained

4.3.1 Simulation

To evaluate how well the ALV homing method works with our type of visual features, a series of simulation experiments were performed first. Here we report the most important findings of these experiments. A more detailed explanation and discussion of the simulation experiments can be found in (Goldhoorn et al, 2007b; Goldhoorn, 2008).

The experiments were done in a simulated environment (see Figure 4.2) with different distributions of feature points. The environment is a room composed of a flat floor, in which the robot moves, and four walls. We have done experiments in this room changing the number of visible walls. The simulated robot was

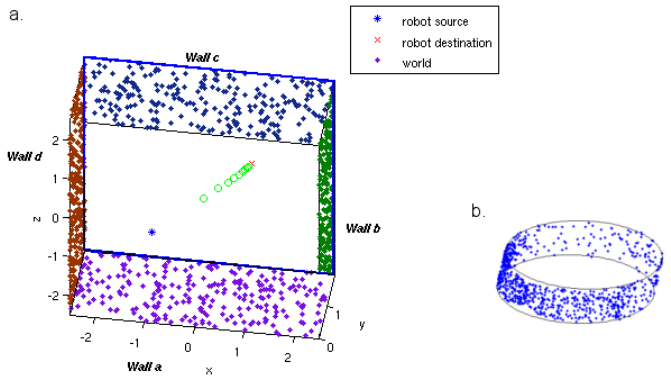


Figure 4.2: a) The simulated environment with uniformly randomly spread feature points. b) Panoramic projection of the world used as input for the robot homing system.

said to be successful if it found the destination point within the following three limitations:

1. The robot is not allowed to use more than 2000 steps (iterations)
2. The projection of the world should not be empty more than five times in a row (in that case either the previous home vector or a vector with random orientation and length was used)
3. The robot should travel at most a distance ten times the Euclidean distance between the start and destination position.

Although the feature points used are robust to most image variations, there are almost always changes due to noise in the localization or occlusions.
Adding Gaussian noise to the positions of the feature points with a standard deviation of 0.001 m or less resulted in a 90% successful runs. However a standard deviation of 0.05 m or more resulted in only 5% or less of successful runs.
Occlusions were simulated by removing randomly chosen feature points before every projection. Removing 50% of the feature points resulted in a mean success rate of 85%. The method was also robust to adding randomly placed feature points, which can be thought of as reappearing previously occluded objects.
Having more reliable feature points present in the world increases the performance of the robot (higher success rate, less iterations and a smaller difference with the ideal distance). For the simulation the range for the number of feature points is between 500 and 1000 for a success rate of 100%. Although having only 20 feature points in the world still resulted in 50% to 80% successful runs. However it has to be taken into account that these runs were without any noise and without any other disturbances.

Because no depth is used, the ALV method implies an equal distance assumption of the landmarks. Franz et al (1998) also mentions the isotropic feature distribution, which can explain why results in a world with only one wall were worse than in the other configurations. The robot used more iterations when more feature points were removed, but this was expected since the ALV every time has a different error.
From these experiments can be concluded that using the ALV for visual homing with visual feature points is a robust method. Therefore the next step was to try this method on a real robot.

4.3.2 IIIA Panoramas Database

This section first explains the experimental setup, then the results are presented and discussed.

Experimental Setup

In these experiments several panorama were acquired at a grid of known points in the rooms. The orientation of the robot was kept constant for each panorama so no alignment step is necessary between the panoramas.
The experiments were done in three rooms of different sizes: the *robot laboratory* , the *square room* and the *corridor*. A scaled map of the rooms can be seen in Figures 4.5, 4.7 and 4.8. Three types of landmarks/feature points were used: 1) DoG feature points; 2) MSER feature points; and, only in the robot laboratory, 3) artificial landmarks.
The locations where the panoramas were created are marked as circles with its identifying number and a line starting at the center of the circle and pointing to the direction of the estimated home vector. The home location is shown as a red circle without line and is also indicated in the figure captions. The biggest objects in the rooms, such as desks, are also shown in the maps to give a rough idea of the environment. Finally, the squares in Figure 4.5 show the landmarks positions and its ID number.
Like in the simulation, only the direction of a feature is known and not its distance, therefore the home vector will not contain distance information either. The home angle calculated by the homing method is compared to the ground truth home angle which is calculated by geometry.

$$\theta_{\text{diff}}(h_{\text{h}}, h_{\text{c}}) = \min\left(|h_{\text{c}} - h_{\text{h}}|; 360 - |h_{\text{c}} - h_{\text{h}}|\right) \tag{4.4}$$

All angles are in degrees and counter-clockwise; h_{c} is the correct homing direction calculated by using the positions (geometry), and h_{h} is computed by the homing method. To find out how well the method works for each room and each type of feature, all the panorama positions per data set are used. For each data set (the *square room*, the *robot laboratory* and the *corridor*) all the locations where a panorama was created are used to calculate the home vector to each of the other locations. From the error calculated with Equation 4.4 for each possible panorama pairings in one room, the mean, median, standard deviation and a

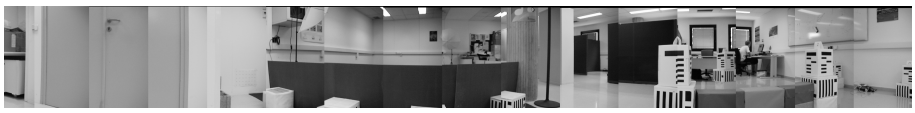


Figure 4.3: Part of a panorama image created by stitching several images together. The image is made in the robot laboratory.

score are calculated. The score is calculated by using the proportion of the maximum error and ranges between 0 and 1 being 1 best. Namely:

$$s = 1 - \frac{\sum_{i=1}^{n} \sum_{j=1;i\neq j}^{n} \theta_{\text{diff}}(h_{\text{h}}(P_i, P_j), h_{\text{c}}(P_i, P_j))}{180n(n-1)} \tag{4.5}$$

where n is the number of panoramas in the set and P the set of panoramas. The numerator is the sum of the difference of the home angle calculated by the ALV homing method and by geometry. This difference, i.e. error, is calculated for each panorama pair, which in total are $n(n-1)$ pairs. The sum of errors is divided by that factor to get an average and, to normalise the score between 0 and 1, it is also divided by 180°, the maximum possible error. The experiments were done in three mentioned different areas in the IIIA research center. The room in which most experiments were done is the robotics laboratory. The panorama in Figure 4.3 shows this room as seen from the robot and in Figure 4.5 a map can be seen. As can be observed in the figure, artificial landmarks are present in the room. These landmarks were used in a set of experiments for comparison purposes with the local feature based approach.

Landmarks

In order to compare our proposed approach to an artificial landmark based one, extra experiments were done using six artificial landmarks in the *robotics laboratory* (see Figure 4.4) available from previous experiments (Busquets, 2003; Busquets et al, 2003).
The landmarks contain a bar code from which an ID number can be extracted. Since the size of the bars is known, the distance to the landmark can be calculated. In order to make the artificial landmark approach comparable to the feature based one, neither the landmark number (for matching) nor the distance information was used in our experiments.

Results

When calculating the home vector between two points, for example a and b, the home vector from a to b will obviously always point in opposite direction of the home vector from b to a. This means that these are dependent values and therefore only one of them was used in the analysis. Next we discuss the results for the three different areas.

Figure 4.4: An example of a landmark in the *robotics laboratory*.

	DoG	MSER	Landmarks
Mean error	35.60°	27.84°	14.88°
Median error	22.85°	16.03°	10.17°
Standard deviation	36.67°	35.51°	14.86°
Score	0.8022	0.8454	0.9173
Best home	117	117	110

Table 4.1: The homing error using the panoramas from the *robotics laboratory*. The *best home* field shows the number of the panorama (see Figure 4.5 for the numbers in the *robotics laboratory*), which when chosen as home, resulted in the lowest average error.

Robotics laboratory: Most panoramas, 38 in total, were acquired in the *robotics laboratory*, a room of 10.5 m × 11.2 m. Only the half of the room is really used for this experiment because the other part is filled with working places and the robot soccer field as can be seen in Figure 4.5.

The home vectors have an error equal to or lower than 90° in 89.3% of the cases when the DoG detector was used, 92.6% for the MSER detector and 99.6% when the landmarks were used. An error of 10° or less was obtained in 22.6% of the cases for the DoG detector, 32.7% for the MSER detector and 64.3% for landmarks. Table 4.1 shows the results for each type of detector used. The homing errors for the three methods are all significantly different ($p < 0.001$) according to the rank sum test, and the t-test after bootstrapping ($n = 1000$). From this can be concluded that the homing method worked best with the artificial landmarks, as expected, and worst with the DoG detector.

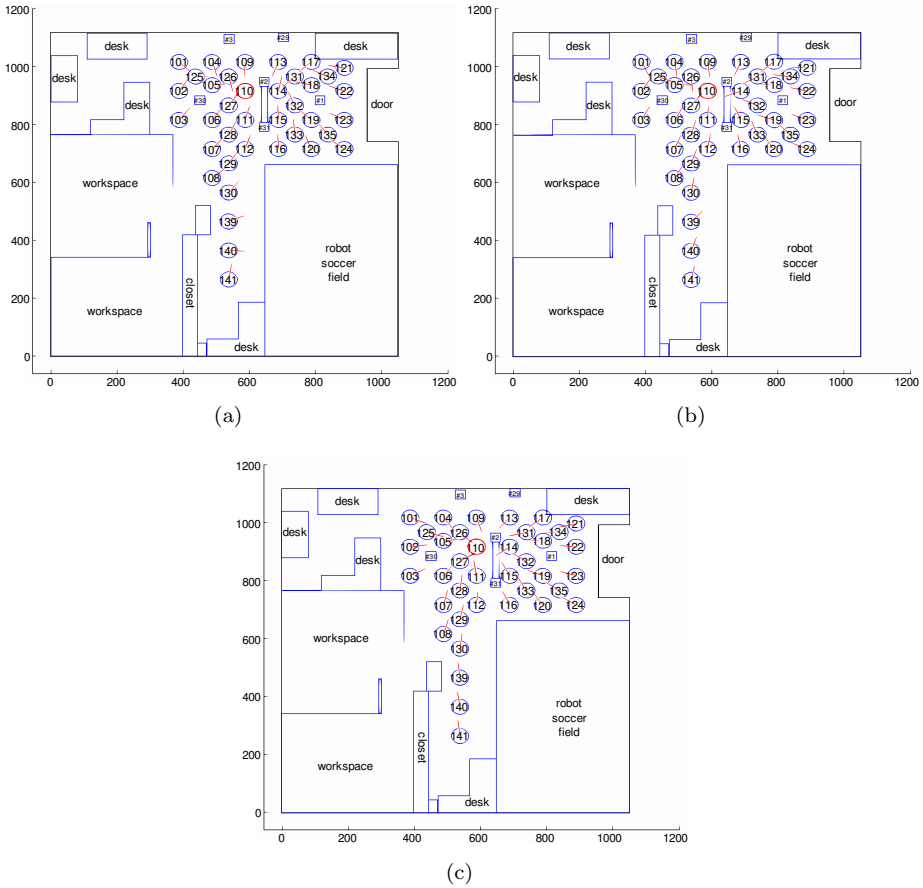


Figure 4.5: Homing to panorama 110 in the *robotics laboratory* using DoG feature points (a), MSER feature points (b) and the landmarks (c). All measures are in cm.

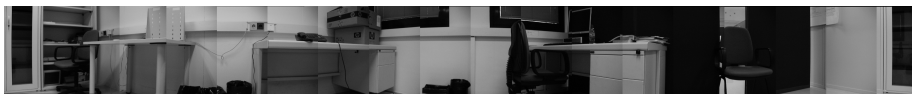


Figure 4.6: Panorama 137 from the *square room*.

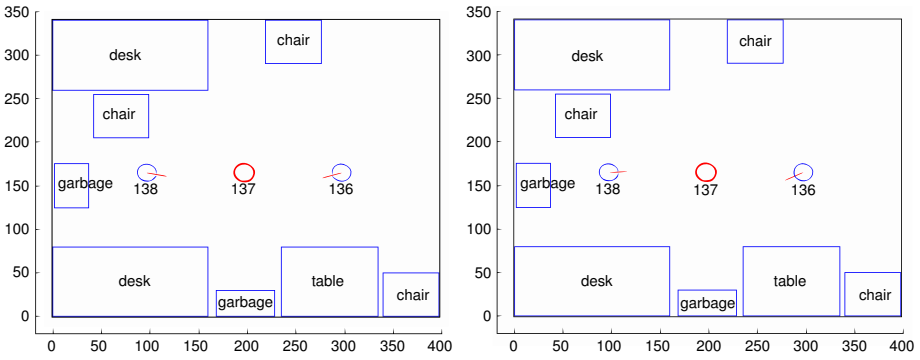


Figure 4.7: Homing to panorama 137 in the *square room* (a) using DoG points and (b) MSER points. All measures are in cm.

Square room: The square room is 4.0 m × 3.4 m big. Figure 4.6 shows a panorama made in this room. Figure 4.7 shows the map of the room and the home vectors to panorama 137. Table 4.2 shows the statistics of the homing method using both feature types. MSER feature points achieved lower error rates than DoG feature points, but this is not significant (confirmed by the rank sum test and the t-test) and it must be noted that only three panoramas were created in this room.

Corridor: Although the simulation showed that the ALV homing method works better in square rooms, we wanted to find out what the impact of a very

	DoG	MSER
Mean error	13.78°	9.65°
Median error	12.00°	12.03°
Standard deviation	11.31°	7.84°
Score	0.9234	0.9464
Best home	138	138

Table 4.2: The error of the homing method using the panoramas which were made in the *square room*.

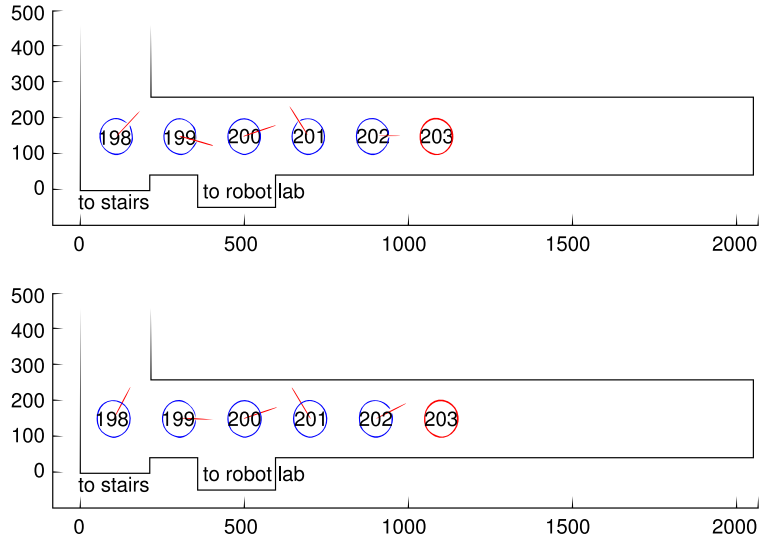


Figure 4.8: Homing to panorama 203 in the *corridor* using (a) DoG feature points and (b) MSER feature points. All measures are in cm.

	DoG	MSER
Mean error	56.26°	52.67°
Median error	44.58°	35.71°
Standard deviation	43.64°	44.90°
Score	0.6874	0.7074
Best home	203	200

Table 4.3: The average error of the homing method in the *corridor* for the different feature types.

long and very narrow room in a real environment would have on the method. A corridor was chosen for that reason as last experiment room. The part of the corridor in which the robot moved is 2.2 m wide and about 22.5 m long. In Figure 4.8 the map of the corridor can be seen. Additionally, Figure 4.9 shows the panoramas acquired in the corridor.
In Figure 4.8 the home vectors to panorama 203 are shown. An error of 90° or less was obtained in 73.3% of the cases for both feature types, an error of 10° or less was only obtained in one case (6.7%). Table 4.3 shows the average error of this data set; the differences between the results with DoG and MSER are not statistically significant.

From the results at the different rooms, it can be seen that the ALV homing method worked better in both the *square room* and the *robotics laboratory*

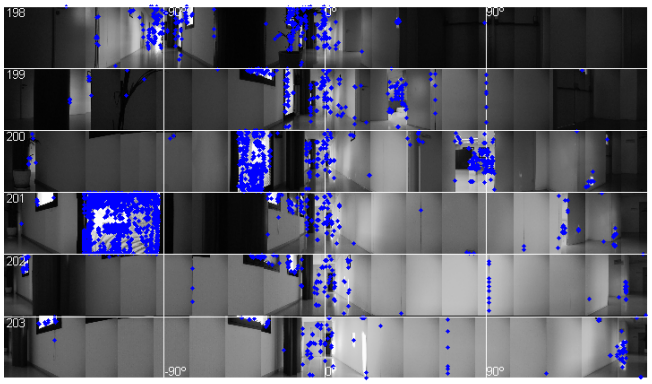


Figure 4.9: All the panoramas made in the corridor. The dots are MSER feature points.

than in the *corridor*. This difference might be explained by the previously found conclusion, in the simulated experiment (Section 4.3.1), that the method works better in approximately square rooms. This is due to the equal distance assumption. In what follows we provide additional details regarding the analysis of the results.

Corridor results: The panoramas acquired in the *corridor* (Figure 4.9) show that there are several *disturbing* factors on which numerous MSER feature points were found. Panorama 198 is the only panorama taken at a corridor intersection, and therefore the MSER detector finds considerable more feature points than in the other panoramas. In panoramas 200 and 201 a door with blinds is visible, and the MSER detector also found a great amount of feature points on these blinds; in panoramas 199 and 200 the robotics laboratory is visible through an open door which again has many feature points. Figures 4.8.a and 4.8.b confirm this, because here the home direction from panoramas 199, 200 and 202 to 203 were good, but from panoramas 198 and 201 really bad. The reason for this is that in panoramas 198 and 201 the most MSER feature points are located on one side only (as commented earlier in this paragraph), while in the other panoramas (199, 200, 202 and the home panorama 203) the feature points are more or less equally distributed. Although only the MSER detector was mentioned here, the DoG detector generated even more feature points, but with a more or less similar distribution.
In Table 4.4 can be seen that the best *corridor* of the IIIA data sets is at rank 25, but this is below the best of the data sets *robot lab* and *square room*.

Upper and lower part: In an attempt to improve the results, the view of the image was limited to only the lower half of the panorama. This part contains objects which are closer to the robot and therefore decrease the size of the visible

world, for this reason a room may look more square.
In the *robotics laboratory* using only the lower half of the panorama resulted in a lower error than using all feature points of the panorama ($p < 0.001$ with the t-test and the rank sum test for both DoG and MSER). For the other rooms there was no significant difference in performance. Also here the best results were when the MSER detector was used ($p < 0.005$ for the *robotics laboratory* and *corridor*) except for the *square room* where DoG was the best detector ($p < 0.001$, rank sum test).
Also the use of only the upper half part of the panorama was tested, but these results were significantly worse than using the whole panorama for the *robotics laboratory* ($p < 0.001$, t-test and rank sum test). There was again no significant difference in the *square room* and *corridor*.

Depth: When only the position of the feature points on the panorama are used then only the direction of the home vector can be found. However, when the distance to the feature points is available, a more precise estimation of the distance to home can be calculated.

4.3.3 Vardy's Panorama Database

As an additional test, we used the image database of Vardy (2005)[1] which he discussed and used to test several homing techniques in his thesis.
Vardy's image database consists of panoramic images acquired over a grid of equally separated points from the hall and the robotics laboratory of Bielefeld University. He created six data sets of the laboratory and two of the hall, all under slightly different conditions, such as the amount of light and added objects. In the robotic laboratory the data set consisted of a 10×17 image grid with 30 cm separation between each image (horizontally and vertically); in the hall 10×21 images in a grid were created per data set with 50 cm separation between images. In contrast to the *IIIA database*, Vardy's database was acquired with an *ImagingSource DFK 4303* camera pointing towards an hyperbolic mirror. This system directly acquires omnidirectional images, and therefore spares the panorama creation step. However it suffers from a much lower resolution. Figure 4.10 shows a panorama from the *hall1* data set. In our experiments, first all the feature points are extracted from the images. As can be seen in Figure 4.10, the image also contains non relevant parts which lay outside the mirror. To focus on the informative area of the image, the field of view is reduced to a limited number of degrees above and below the horizon, which is the line between the centre of the spherical mirror and the outer circle of the mirror. Only feature points which fall in this area are used for the homing method.
The vector of a feature has its origin in the image centre (shown as the red dot in Figure 4.10) and points to the feature point. These vectors have to be normalised to 1 before calculating the ALV, because the length of the vectors

[1] Vardy's *Panoramic Image Database* is available at
`http://www.ti.uni-bielefeld.de/html/research/avardy/index.html`.

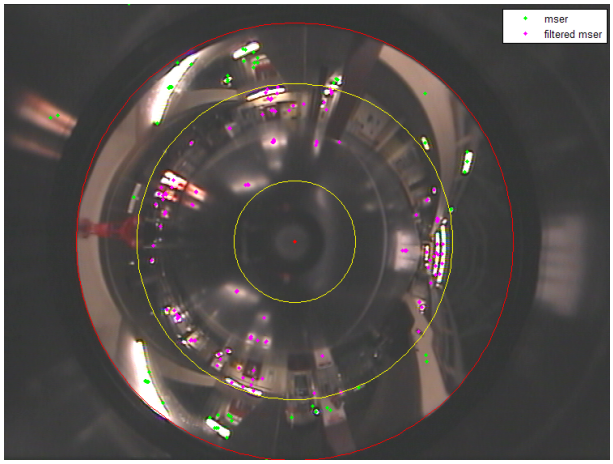


Figure 4.10: A panorama from Vardy's image database. The outer red circle shows the border of the parabolic mirror, the two inner yellow circles show the 20° line above and below the horizon. The points show the location of the MSER feature points; the filtered feature points are the ones between the yellow circles (best viewed in color).

only show the distance in pixels on the image. After this, the ALVs and the home vector can be calculated as described in section 4.1.
Table 4.4 shows the results with all data sets sorted by score.

Results

As can be seen from the Table 4.4, the scores for Vardy data set vary from 0.85 to 0.3 and the home angle error from $28.2° \pm 27.6$ to $126.0° \pm 43.3$. The results are worse than the results with the previously discussed data sets, but it must be noticed that Vardy's data sets contain more samples.
It can be seen that a wider vertical view angle gives better results. When MSER feature points were used, a view angle of 15° (above and below the horizon) worked significantly better than a lower angle ($p < 0.001$, t-test and rank sum test). For all data sets except for *doorlit* and *hall1* the best view angle was 20°. This is also the case when DoG feature points were used, except for the data sets *day*, *hall2* and *screen*. In the data set *day* the difference was not significant enough; using a view angle of 5° had the best results in the sets *hall2* ($p < 0.001$, rank sum test) and *screen* ($p < 0.05$, rank sum test) when DoG feature points were used.
It is also clear from Table 4.4 that the performance is better when using the MSER detector than the DoG detector. This difference is significant for all data sets with a view of more than 5° above and below the horizon (using the t-test and rank sum test; $p < 0.001$). It also can be seen from the table that the

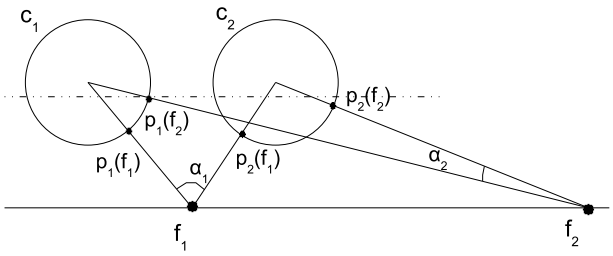


Figure 4.11: The position where points are projected in different panoramas varies less (and therefore are less informative) if the points are far away. This is a problem for narrow and long corridors with most texture at the extremes.

best of the IIIA sets are all above the data sets of Vardy, however this is only significant for the *robotics laboratory*. Comparing the results of Vardy's data set with the results of the IIIA data set is difficult because of several reasons. First of all there are two big differences between them: the environments and the way the panoramas are made. The rooms could be assumed to be quite similar since they both are flat "office like" with several desks, chairs and computers, but the landmarks present in the *robotics laboratory* are not present in Vardy's rooms for example.

4.4 Overall Discussion of Experimental Results

As has been seen, with the real robot experiments the ALV homing method gave very positive results. The best results were obtained with the panoramas from the *square room*. The results from the *corridor* were worst, as expected. In the simulation already was found that the performance of the homing method is better in square rooms than in rooms with big differences in width and length. The problem with long rooms such as a corridor is that the projections of the feature points onto a panorama are closer to each other the further away they are from the robot (see Figure 4.11).
Looking at the difference in performance using DoG and MSER feature points it can be concluded that the use of MSER feature points significantly outperforms the use of DoG feature points. The artificial landmarks in the *robotics laboratory* were used to find out how well the method worked in comparison with invariant feature points. The results with the artificial landmarks were significantly better than using invariant feature points, the error was about 7° less than using MSER feature points (with only the lower half of the panorama).
Normally one should expect the homing method to work worse when the distance between the current location and the home is lower, but this relation could not be found. This might be because the room is too small or because objects occlude a big part of the field. Further work would be needed to find out if there is any

relation between the distance and error.
An attempt to improve the results was done by trying to make the rooms, such as the corridor, more square by only using the lower half of the panorama, because then the closer objects are more prominent. This however had no significant improvement in the *corridor*, and neither in the *square room*. Only in the *robotics laboratory* there was a significant lower error ($p < 0.001$).
The images of Vardy (2005) data sets were also used to test the ALV homing method. Although the different panorama acquisition system, in practice the performance of these sets was not much worse than the results of the IIIA ones. From these images also SIFT and MSER feature points were extracted and used to calculate the ALV. It was found that using almost the whole image (20° above and below the horizon) resulted in the best performance.
The scores (with 1 being best and 0 begin worst) of the IIIA data sets varied from 0.67 to 0.96, whereas the results of Vardy's data sets varied from 0.30 to 0.85 (see Table 4.4). Looking at the best parameters however, such as using the lower half of the panorama for the IIIA data sets and using a view angle of 20° above and below the horizon of Vardy's data, the scores of the IIIA data sets vary from 0.73 to 0.96 and the scores of Vardy's data sets from 0.67 to 0.85. This shows that the method performs almost as well in the different rooms and with the different types of panoramas, and thereby confirms the robustness of the method.
Finally some comparison to other work can be made, however in most works other error measurements are used such as the distance at which it stops from home. In this work however no such experiments have been done yet. Hafner (2001) also did experiments in an office environment in a grid. After off-line learning the average error was smaller than 90° in 92% of the cases and smaller than 45° in more than 69%. This is comparable to the results in the *robotics laboratory* for the DoG feature points, and our results for using MSER feature points were even better. The experiments by Franz et al (1998) were done in a 118 cm × 110 cm environment but the catchment area was relatively smaller than the catchment area of the IIIA data sets. Their algorithm performed robustly up to an average distance of 15 cm. They also mention experiments done in an office environment in which the algorithm performed robustly until about 2 m.

#	dataset	type	detector	mean	median	std.dev.	score	bh	n
1	square room	upper half	MSER	6,83	4,10	5,33	0,9621	138	3
2	square room	not filtered	MSER	9,65	12,03	7,84	0,9464	138	3
3	square room	not filtered	DoG	13,78	12,00	11,31	0,9234	138	3
4	robot lab	not filtered	Landmarks	14,88	10,16	14,86	0,9173	110	38
5	square room	lower half	DoG	14,94	14,52	10,75	0,9170	138	3
6	square room	lower half	MSER	20,62	25,27	8,49	0,8855	138	3
7	square room	upper half	DoG	20,91	18,96	6,64	0,8838	138	3
8	robot lab	lower half	MSER	21,96	11,09	30,05	0,8780	117	38
9	day	20	MSER	26,18	18,73	27,62	0,8545	17	170
10	robot lab	lower half	DoG	26,90	13,05	34,74	0,8506	117	38
11	robot lab	not filtered	MSER	27,84	16,03	35,51	0,8454	117	38
12	screen	20	MSER	28,64	18,42	31,04	0,8409	95	170
13	doorlit	15	MSER	30,69	19,35	33,38	0,8295	15	170
14	arboreal	20	MSER	34,89	23,39	35,49	0,8061	50	170
15	doorlit	20	MSER	35,41	21,27	38,52	0,8033	50	170
16	robot lab	not filtered	DoG	35,60	22,85	38,67	0,8022	117	38
17	arboreal	15	MSER	37,83	25,31	37,20	0,7898	17	170
18	day	15	MSER	39,78	29,30	36,49	0,7790	17	170
19	hall1	15	MSER	42,61	31,55	38,32	0,7633	159	200
20	original	20	MSER	43,18	32,23	38,49	0,7601	50	170
21	screen	15	MSER	45,71	33,75	40,43	0,7461	0	170
22	hall1	10	MSER	45,81	35,12	39,05	0,7455	41	200
23	twilight	20	MSER	46,21	34,90	39,72	0,7433	50	170
24	doorlit	10	MSER	48,45	33,95	43,90	0,7308	14	170
25	corridor	lower half	MSER	48,83	39,02	41,63	0,7287	203	6
26	arboreal	10	MSER	49,97	35,14	44,80	0,7224	153	170
27	robot lab	upper half	MSER	50,62	39,33	42,47	0,7188	117	38
28	winlit	20	MSER	52,39	39,58	44,07	0,7089	50	170
29	corridor	not filtered	MSER	52,66	35,71	44,89	0,7074	200	6
30	robot lab	upper half	DoG	56,14	45,77	43,84	0,6881	117	38
31	corridor	not filtered	DoG	56,26	44,58	43,64	0,6874	203	6
32	corridor	lower half	DoG	56,45	49,69	42,39	0,6864	203	6
33	corridor	upper half	DoG	57,08	38,19	45,65	0,6829	203	6
34	twilight	15	MSER	57,63	44,59	46,70	0,6798	153	170
35	hall1	20	MSER	58,49	48,71	44,53	0,6751	99	200
36	original	15	MSER	58,53	45,11	47,56	0,6748	17	170
37	chairs	20	MSER	58,92	45,23	47,55	0,6726	84	170
38	corridor	upper half	MSER	59,15	42,56	46,02	0,6714	199	6
39	hall2	20	MSER	59,51	49,20	43,09	0,6694	18	200
40	hall2	15	MSER	61,00	50,90	45,30	0,6611	18	200

Continued

#	dataset	type	detector	mean	median	std.dev.	score	bh	n
41	day	10	MSER	62,50	51,30	47,17	0,6528	17	170
42	hall1	5	MSER	62,69	53,82	44,78	0,6517	39	200
43	screen	10	MSER	63,43	54,30	45,02	0,6476	153	170
44	screen	5	MSER	66,71	52,61	50,03	0,6294	102	170
45	hall2	5	MSER	68,57	57,22	49,19	0,6190	19	200
46	original	10	MSER	72,09	62,67	50,85	0,5995	14	170
47	winlit	15	MSER	72,39	63,58	50,84	0,5978	50	170
48	day	5	MSER	72,67	62,94	50,31	0,5963	169	170
49	hall2	10	MSER	72,72	62,01	50,68	0,5960	19	200
50	twilight	10	MSER	73,55	65,58	51,23	0,5914	18	170
51	hall1	20	DoG	77,16	71,90	45,43	0,5713	0	200
52	winlit	10	MSER	78,69	72,12	51,96	0,5628	16	170
53	chairs	15	MSER	79,61	72,79	51,91	0,5577	16	170
54	doorlit	5	DoG	80,15	75,59	52,15	0,5547	169	170
55	doorlit	20	DoG	83,07	80,38	49,17	0,5385	151	170
56	doorlit	10	DoG	83,79	81,13	50,27	0,5345	169	170
57	chairs	10	MSER	84,42	82,22	50,74	0,5310	14	170
58	doorlit	15	DoG	84,44	81,34	49,75	0,5309	152	170
59	chairs	20	DoG	86,15	82,29	49,48	0,5214	3	170
60	screen	5	DoG	86,87	85,63	51,82	0,5174	135	170
61	screen	15	DoG	88,42	85,67	51,75	0,5088	135	170
62	screen	10	DoG	88,76	88,36	52,02	0,5069	135	170
63	screen	20	DoG	89,25	86,89	51,88	0,5041	3	170
64	hall1	15	DoG	89,27	85,64	45,78	0,5041	0	200
65	chairs	15	DoG	90,33	87,09	51,16	0,4981	4	170
66	arboreal	20	DoG	90,33	88,88	50,27	0,4981	4	170
67	original	20	DoG	91,36	88,78	49,70	0,4924	3	170
68	twilight	20	DoG	91,66	89,34	49,59	0,4908	5	170
69	day	10	DoG	92,99	94,81	51,31	0,4834	152	170
70	day	15	DoG	93,00	94,20	50,95	0,4833	135	170
71	day	20	DoG	93,05	93,13	50,42	0,4830	135	170
72	day	5	DoG	93,10	94,15	51,72	0,4828	152	170
73	chairs	10	DoG	93,46	92,24	51,88	0,4808	4	170
74	chairs	5	DoG	93,51	92,00	50,99	0,4805	5	170
75	arboreal	15	DoG	95,20	95,05	51,65	0,4711	4	170
76	twilight	15	DoG	96,44	96,35	50,29	0,4642	5	170
77	winlit	5	MSER	97,11	103,58	54,36	0,4605	136	170
78	original	15	DoG	97,93	97,66	49,81	0,4559	4	170
79	winlit	20	DoG	98,86	99,65	44,11	0,4508	0	170
80	arboreal	10	DoG	99,07	101,86	51,64	0,4496	135	170
81	arboreal	5	DoG	100,55	104,77	51,59	0,4414	135	170
82	twilight	10	DoG	101,25	102,97	50,17	0,4375	6	170
83	hall1	10	DoG	101,86	99,59	44,55	0,4341	40	200
84	original	10	DoG	101,98	105,32	50,19	0,4335	4	170
85	twilight	5	DoG	102,16	105,37	49,95	0,4324	6	170
86	doorlit	5	MSER	102,79	110,41	51,90	0,4290	14	170
87	winlit	5	DoG	103,01	109,13	46,78	0,4277	134	170
88	original	5	DoG	103,19	108,47	50,36	0,4267	50	170
89	winlit	15	DoG	103,28	105,54	45,33	0,4262	34	170
90	winlit	10	DoG	104,93	109,14	46,08	0,4171	135	170
91	hall1	5	DoG	108,85	109,20	46,09	0,3953	80	200
92	arboreal	5	MSER	112,73	122,76	48,51	0,3737	14	170
93	chairs	5	MSER	116,47	126,06	46,16	0,3529	14	170
94	hall2	5	DoG	116,47	130,14	49,96	0,3529	198	200
95	original	5	MSER	118,50	128,27	44,93	0,3416	153	170
96	hall2	20	DoG	122,09	132,21	44,38	0,3217	20	200
97	twilight	5	MSER	122,21	133,26	44,17	0,3211	136	170
98	hall2	10	DoG	124,42	137,90	45,31	0,3088	199	200
99	hall2	15	DoG	125,99	137,57	43,33	0,3000	61	200

Continued

#	dataset	type	detector	mean	median	std.dev.	score	bh	n

Table 4.4: This table shows the results of all real world experiments (IIIA in white and Vardy in light gray) sorted by score. For the IIIA dataset, the type column shows which part of the panorama has been used: all feature points (*not filtered*), only the feature points at the *lower half* of the panorama or only at the *upper half* (see Section 4.3.2). For Vardy data set, the type column shows the number of degrees above and below the horizon of the image which were used. The detector column shows which feature detector has been used to perform homing: DoG, MSER or artificial *landmarks* which were only available in the *robot laboratory*. The next three columns: mean, median and std. dev. (standard deviation) show information about the direction error of the home vector in degrees. The calculation of the score is shown in Equation 4.5; 1 being best and 0 being worst. The *best home* (bh) column shows the ID of the location of the home where to the mean error is smallest. Finally the n column shows the number of samples, i.e. different panoramas, for the data set.

Chapter 5

SIFT Object Recognition Method

Lowe's SIFT object recognition approach is a view-centered object detection and recognition system with some interesting characteristics for mobile robots, most significant of which is the ability to detect and recognize objects at the same time in an unsegmented image. Another interesting feature is the Best-Bin-First algorithm used for approximated fast matching, which reduces the search time by two orders of magnitude for a database of 100,000 keypoints for a 5% loss in the number of correct matches (Lowe, 2004).

The first stage of the approach consists on matching individually the SIFT descriptors of the features detected in a test image to the ones stored in the object database using the Euclidean distance. False matches are rejected if the distance of the first nearest neighbor is not distinctive enough when compared with that of the second. In Figure 5.1.b, the matching features between a test and model images can be seen. The presence of some outliers can also be observed. Once a set of matches is found, the generalized Hough Transform is used to cluster each match of every database image depending on its particular transformation (translation, rotation and scale change). Although imprecise, this step generates a number of initial coherent hypotheses and removes a notable portion of the outliers that could potentially confuse more precise but also more sensitive methods. All clusters with at least three matches for a particular training object are accepted, and fed to the next stage: the Least Squares method, used to improve the estimation of the affine transformation between the model and the test images.

This approach has been modified in several ways in our experiments. The breakdown point (i.e. ratio of outliers in the input data were the model fitting method fails) for the least squares method is at 0% of outliers, which is a rather unfeasible restriction since we have found it is normal to still have some false matches in a given hypothesis after the Hough Transform. To alleviate this, instead of the least squares, we have used the Iteratively Reweighted Least Squares (IRLS).

(a) (b)

Figure 5.1: (a) Diagram of the modified SIFT object recognition method. (b) Matching stage in the SIFT object recognition method.

Furthermore we have added the RANdom SAmple Consensus (RANSAC), another well-known model fitting algorithm that iteratively tests the support of models estimated using minimal subsets of points randomly sampled from the input data. Finally, we have manually defined a set of heuristic rules on the parameters of the estimated affine transformation to reject those clearly beyond plausibility. Namely:

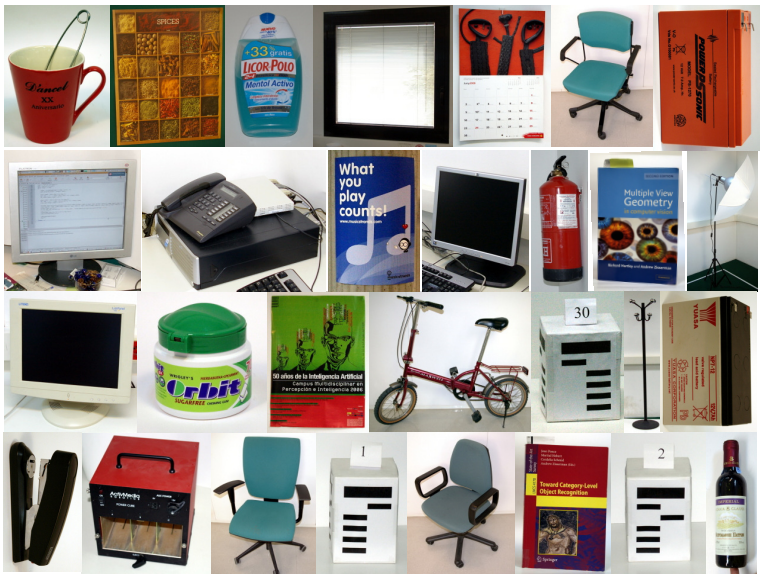
- Hypotheses for repeated objects with too close centers.
- Hypotheses that have a ratio between the x and y scales below a threshold.

Figure 5.1.a shows an overview of our implementation of the SIFT object recognition algorithm steps.

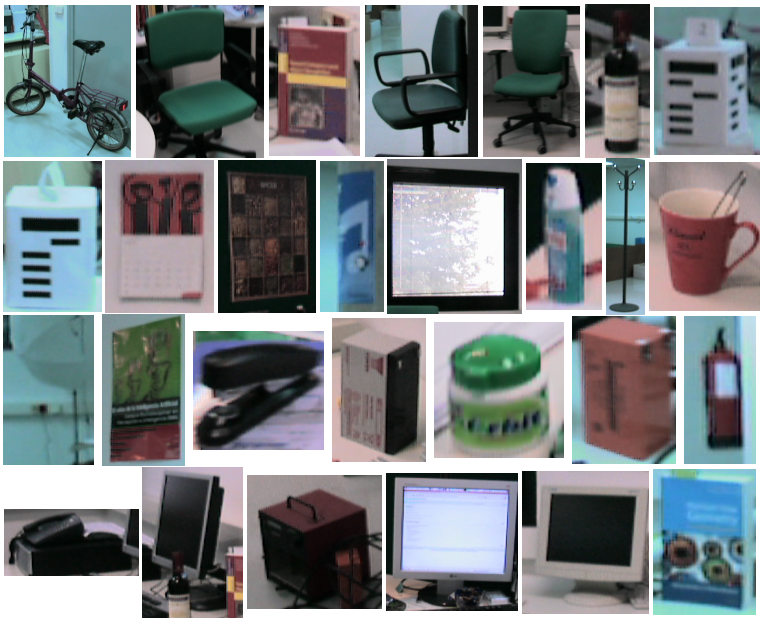
5.1 IIIA30 Database

In order to evaluate the methods in a realistic mobile robots setting, we have created the IIIA30 database[1], that consists of three sequences of different length acquired by our mobile robot while navigating in a laboratory type environment. Image size is 640x480 pixels. The environment has not been modified in any way and the object instances in the test images are affected by lightning changes, blur caused by the motion of the robot, occlusion and large scale and viewpoint changes. We have considered a total of 30 categories (29 objects and background) that appear in the sequences. The objects have been selected to cover a wide range of characteristics: some are textured and flat, like the posters, while others are textureless and only defined by its shape. Figure 5.2.a shows the training images for all the object categories, and 5.2.b shows some cropped object instances from the test images. Each occurrence of an object in the video

[1] `http://www.iiia.csic.es/~aramisa/iiia30.html`



(a)



(b)

Figure 5.2: (a) Training images for the IIIA30 dataset. (b) Cropped instances of objects from the test images.

sequences has been manually annotated in each frame to construct the ground truth, along with its particular image characteristics (e.g. blurred, occluded...).

5.2 Parameter Tuning

In order to find the best set of parameters for the SIFT object recognition system, a series of experiments were done with the IIIA30 dataset. Each experiment aims to evaluate a particular aspect of the method. In most cases, the f-measure has been used to compare the performance of each parameter combination.

$$f - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5.1}$$

This measure weights equally precision and recall and is also known as $f_1 - measure$ or balanced $f - score$. The general f-measure is defined as:

$$f_g - measure = \frac{(1 + \beta^2) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{5.2}$$

Where the β parameter can be used to adjust sensitivity to precision and recall of the measure.
Although the f-measure is not the most standard error measure in object recognition, we have used it here since it allows to assign a clear score to each parameter combination in a principled way. Nevertheless, as not all situations tolerate both error types equally, we also discuss precision and recall individually where possible. Detailed results are additionally provided online for the interested reader[2]. Nonetheless, speed is probably the most relevant performance measure in our setting, and therefore we search for the parameter combinations that perform as close as possible to real-time while retaining a good precision and recall.
To consider an object as a true positive, the bounding boxes of the ground truth and the detected instance must have a ratio of overlap equal or greater than 50% according to the following equation:

$$\frac{BB_{gt} \cap BB_{detected}}{BB_{gt} \cup BB_{detected}} \geq 0.5 \tag{5.3}$$

where BB_{gt} and $BB_{detected}$ stand for the ground truth and detected object bounding box respectively. For objects marked as occluded only the visible part will be annotated in the ground truth, while the SIFT object recognition method will still try to adjust the detection bounding box for the whole object based in the visible part. For the case of occluded objects, we have therefore modified the above formula in the following way:

$$\frac{BB_{gt} \cap BB_{detected}}{BB_{gt}} \geq 0.5 \tag{5.4}$$

As can be seen in the previous equation, it is only required that the detected object bounding box overlaps 50% of the ground truth bounding box. What follows is a detailed discussion of the results obtained for every parameter dimension.

[2] `http://www.iiia.csic.es/~aramisa/iiia30.html`

Feature Detectors and Descriptor: A handful of feature detectors have been proposed in the literature that find different structures in images. These feature detectors vary in number of detected regions and robustness to image variations and, a priori, it is difficult to chose one or a combination of various among the available options. We have evaluated seven feature detectors: Harris Affine, Hessian Affine, Harris Laplace, Hessian Laplace, MSER[3], SURF[4] and DoG[5]. We have used the Oxford SIFT implementation[6] to compute the descriptor of feature regions detected with the first six feature detectors, while the descriptors for the DoG regions have been computed with Lowe's original implementation of SIFT that comes with the DoG detector. It is important to understand that both implementations give significantly different results as can be appreciated in Figure 5.3. As our objective is to obtain the best results and compare the object recognition methods, we have used the best performing implementation with the DoG features and, as it was not possible to use it with other detectors, the Oxford implementation with the rest.

As can be seen in Figure 5.4.a, Hessian based detectors (Hessian Affine and Hessian Laplace) obtained the highest recall but also suffered from a low precision. Harris-based detectors obtained results on the line of the Hessian-based ones, but with a slightly lower recall and precision. Overall, the best f-measure has been obtained by the DoG detector followed by SURF. Finally, the MSER detector had a very low recall. The explanation for these results seems to be in the number of features found by each detector (see Figure 5.4.b). Harris and Hessian based detectors find enough features to achieve high recall rates, but without additional filtering of hypotheses, precision drops below 10%. Furthermore, the computational cost of matching the features and processing the hypotheses increases notably. On the other hand, the MSER detector finds very discriminative

[3]http://www.robots.ox.ac.uk/ vgg/research/affine/detectors.html
[4]`http://www.vision.ee.ethz.ch/~surf/`
[5]`http://www.cs.ubc.ca/~lowe/keypoints/`
[6]`http://www.robots.ox.ac.uk/~vgg/research/affine/descriptors.html`

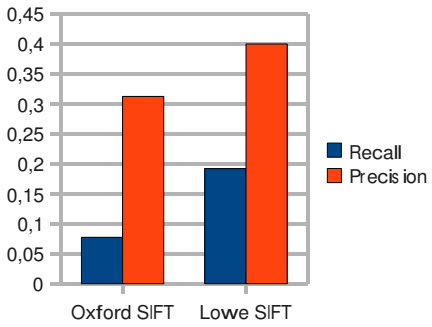
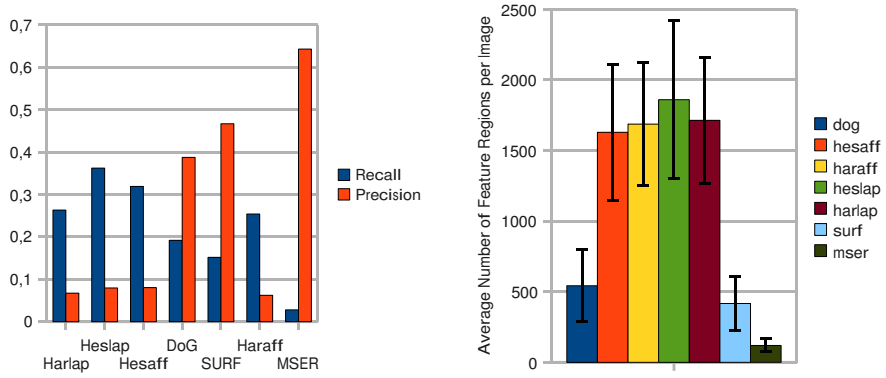


Figure 5.3: Results obtained with the two SIFT descriptor implementations using the DoG feature detector.



(a) (b)

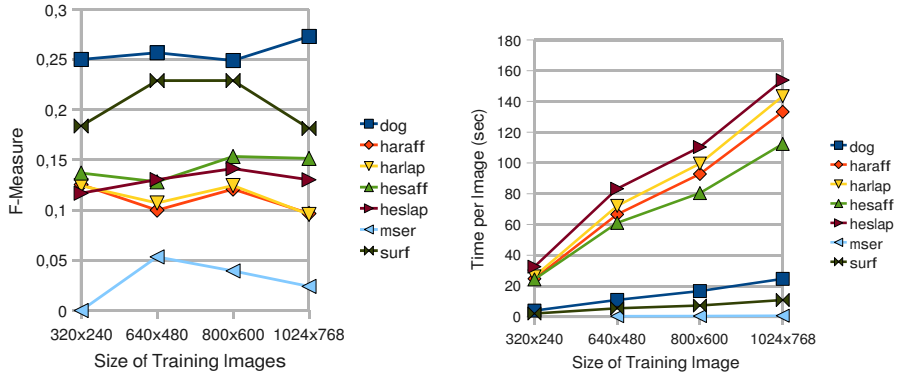
Figure 5.4: (a) Precision and recall depending on feature type (640x480 pixels training images). (b) Average detected feature regions per image in testing data.

features but not sufficient to recognize most of the object instances. The best compromise is achieved by DoG and SURF. Additionally to feature detectors, we have considered the Shape Context descriptor (Mori et al, 2005), but results were not competitive and therefore are not displayed.

Training Image Size: The original SIFT object recognition is designed to be a one shot object recognition method, which makes the choice of the training image an important decision. In our experiments we have considered both: images extracted from a sequence acquired with the robot cameras –to enhance the similarity between the training and the test data– and good quality images of the objects acquired with a conventional digital camera to maximize the number of detected regions. Training images selected from a different sequence acquired with the robot did not have a competitive result, so they were discarded. For the good quality images, we have considered four different image sizes: 320x240, 640x480, 800x600 and 1024x768 pixels.

Figure 5.5.b shows that the time spent in the matching process increases significantly with the training image size, that was expected as the number of detected features also increases. Contrarily, the f-measure did not improve significantly with the increase of training image size. The cause of the erratic behavior of the f-measure is that, although more true positives were found with bigger training images, the number of false positives increased as well.

As no clear advantage was observed in using larger training images, we fixed the 640×480 size for the remaining experiments. This image size is a good compromise between speed and results and, as can be seen in Figure 5.4.b, 320×240 was not sufficient for MSER, as it was not able to find enough features.



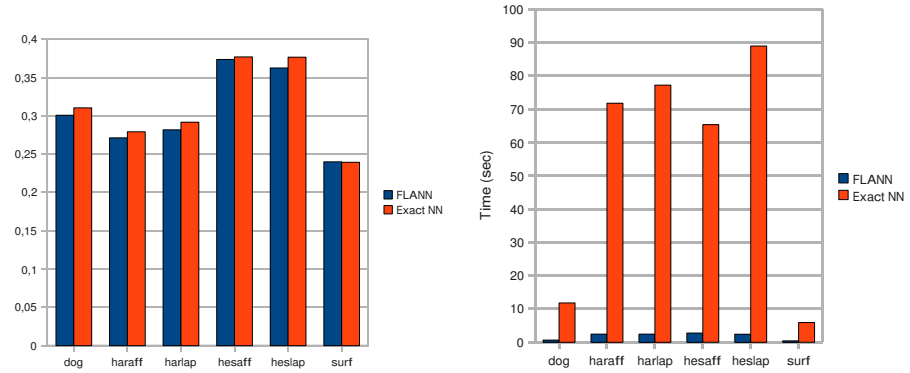
(a) (b)

Figure 5.5: (a) F-Measure depending on the training image size. (b) Time per image depending on training image size with exact nearest neighbor matching.

Matching Method: Various approximate nearest neighbors alternatives have been proposed in the literature (Lowe, 2004; Muja and Lowe, 2009; Lepetit et al, 2004) in order to accelerate the matching process between feature descriptors. As mentioned before, in the original article of the SIFT object recognition algorithm a K-D tree was used with the Best-Bin-First algorithm. Later Muja and Lowe (2009) proposed an improved approach, coined FLANN, which we compare with exact nearest neighbor matching. As can be seen in Figure 5.6, the approximate nearest neighbors method drastically improves the time per image without affecting significantly the performance.

Distance Ratio: The distance ratio between the first and the second nearest neighbor required to accept a match is a critical choice, as it will directly influence the amount of false positive hypotheses generated (and consequently processing time) if too permissive, and the recall if too restrictive. In the original SIFT object recognition approach, the distance ratio between the first and the second nearest neighbor was required to be inferior to 0.8 in order to accept a match. However, as can be seen in Figure 5.7.a we found that different feature types have different optimal values for this threshold: for the Hessian and Harris based detectors, the best value for f-measure is 0.6, while DoG attains the best results at 0.7 and SURF at 0.8. As can be seen in Figure 5.7.b, time spent in the Hough Transform and IRLS stages increases rapidly as more potentially false matches are accepted. Keeping in mind that our aim is producing good enough results within tight time constraints, the choice of a restrictive distance ratio seems attractive.

Hough Transform: As in Lowe's SIFT object recognition method each match votes for 16 bins in the Hough Transform, multiple neighboring bins can easily



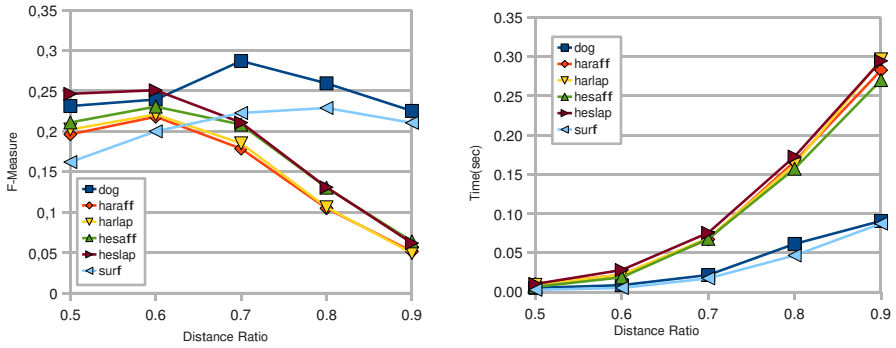
(a) (b)

Figure 5.6: (a) F-Measure depending on the matching method. (b) Time per image depending on matching method.

	DoG			Hessian Affine			DoG2		
	HT	IRLS	Pre	HT	IRLS	Pre	HT	RIH	Pre
Standard	14 ms	11 ms	0.14	27 ms	229 ms	0.02	16 ms	84 ms	0.82
NMS	56 ms	5 ms	0.40	89 ms	68 ms	0.08	73 ms	48 ms	0.87

Table 5.1: Three experiments with the two different Hough Transform approaches: Standard and with non-maxima suppression. The first two columns of each experiment show the time spent in the Hough clustering and in the hypotheses refinement stages respectively, and the third column shows the precision achieved (recall varies at most 0.01 between both HT approaches). In the third parameter combination, RIH stands for the combination of RANSAC, IRLS and Heuristics filtering stages.

be activated for the same object, leading to false or *shadow* hypotheses that consume processing time in successive stages to end up being finally rejected or, even worse, generating false positives. To alleviate this we evaluated the effect of introducing a non-maxima suppression (NMS) step to the Hough Transform. Table 5.1 shows the results of three different experiments with both the standard and the NMS approaches. In the standard configuration with DoG features, the NMS step does not pay off in terms of computational complexity, but increases significantly the precision. However, if the number of false matches is high such as in the case of Hessian Affine with a 0.8 distance ratio, the time savings of the IRLS step are considerable. In the last experiment, additional hypothesis filtering steps are added in order to raise the precision of the standard approach to a value similar to that of the NMS. However, this extra steps increase the time to a similar value also.



(a) (b)

Figure 5.7: (a) F-Measure depending on the distance ratio. (b) Time spent in the Hough Transform and IRLS depending on the distance ratio.

Minimum number of votes in the Hough Transform bins: Although three matches are sufficient to estimate the pose of an object up to an affine transformation, often this number of points is low enough to be a product of chance and the introduced hypotheses will have to be discarded in subsequent steps. In spite of the theoretical justification, empirically we did not find much advantage on increasing the minimum number of feature points regarding performance as can be seen in Figures 5.8.a and 5.8.c. Precision did not increase significantly and recall degraded after losing hypotheses with few support. However, regarding computational cost, increasing the minimum number of features decreased time spent in both Hough clustering (there were less putative bins to perform non-maxima suppression) and hypotheses refinement stages (less hypotheses to verify) as shown in Figures 5.8.b and 5.8.d.

Hypotheses Verification and Refinement: After the clustering of the found matches in the Hough Transform bins, the candidate object hypotheses are subject to a pose estimation up to an affine transformation with an iterative least squares method. This step also reduces the number of false positives by discarding those whose support falls below the minimum number of matches specified (three by default). We evaluated the impact of introducing other robust model fitting and filtering methods to discard a higher number of false positives. Specifically we used, in addition to the Iterative Reweighted Least Squares (IRLS), the RANdom SAmple Consensus (RANSAC) and a set of manually defined heuristics on the detected object bounding box to eliminate repetitions and hypotheses which described unrealistic transformations. As can be seen in Figure 5.9.a, the f-measure increases as more strict filtering methods are applied. The best result is obtained combining all the filtering methods with the Hessian-based feature detectors. This is not surprising as these detectors obtained the best recall but suffered from a high number of false positives. Adding

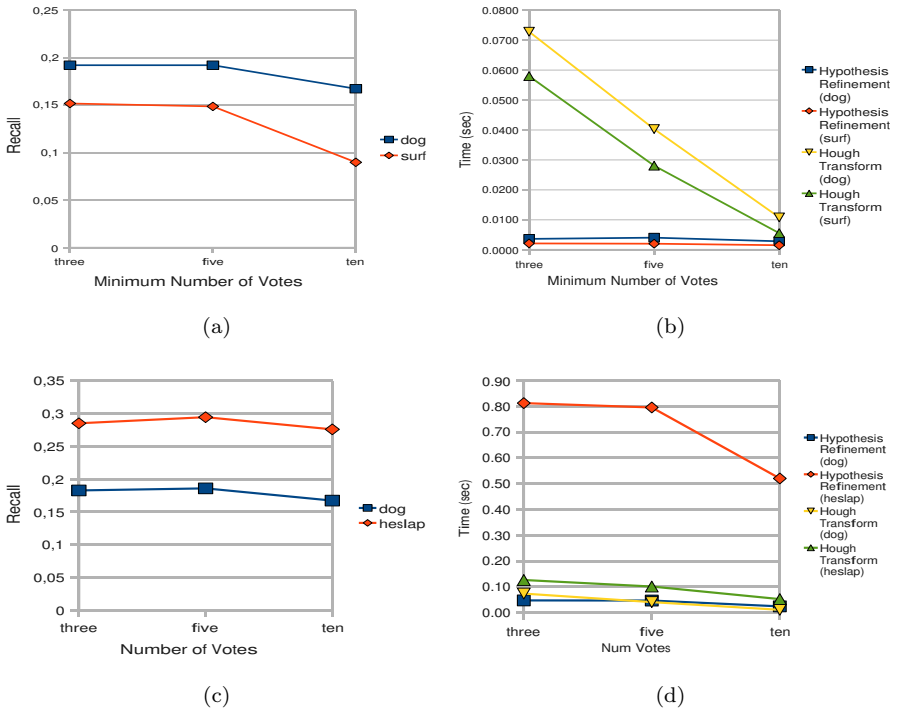
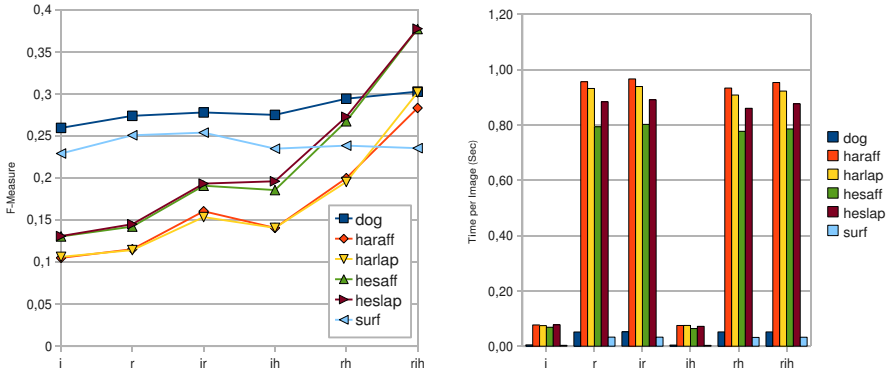


Figure 5.8: Performance depending on minimum number of votes in a Hough Transform bin to accept an hypothesis. (a-b) Shows results for the DoG and SURF detectors with only IRLS hypothesis filtering stage. (c-d) Shows results for DoG and Hessian Laplace detectors with all hypothesis filtering methods proposed (IRLS, RANSAC and Heuristics).

better hypotheses verification methods the precision and therefore the f-measure are improved. The false positives that IRLS alone is not able to filter are mainly due to untextured or repetitively textured objects. The major drawback of these extra methods is an increase of the processing time in the hypotheses verification stages, especially in the case of RANSAC due to its Monte Carlo nature.

5.2.1 Discussion and Selected Configurations

In this section the results of sequence 1 of the IIIA30 dataset (IIIA30-1) with the different parameter combinations considered are evaluated. Taking into account all combinations, the best recall obtained has been 0.45 with the Hessian Laplace detector and the less restrictive settings possible. However this configuration suffered from a really low precision, just 0.03.
The best precision score has been 0.94, and has been obtained also with the Hessian Laplace detector, with a restrictive distance ratio to accept matches:



(a) (b)

Figure 5.9: (a) F-Measure depending on the hypotheses filtering methods and (b) time spent in the filtering stage per image. i stands for IRLS, r for RANSAC and h for heuristics.

Method	Distance Ratio	Detector	Min. Matches	HT Method	RANSAC	Approx-NN	IRLS	Heuristics
Config 1	0.8	SURF	5	NMS	No	Yes	Yes	No
Config 2	0.8	SURF	3	NMS	Yes	Yes	Yes	Yes
Config 3	0.8	DoG	10	NMS	No	Yes	Yes	No
Config 4	0.8	DoG	10	NMS	Yes	Yes	Yes	Yes
Config 5	0.8	DoG	5	NMS	Yes	Yes	Yes	Yes
Config 6	0.8	HesLap	10	NMS	Yes	Yes	Yes	Yes

Table 5.2: Detailed configuration parameters for the six chosen configurations in increasing time order.

0.5. The recall of this combination was 0.14. The same precision value but with lower recall has been obtained with the SURF and Hessian Affine detectors.

Looking at the combinations that had a best balance between recall and precision (best f-measure), the top performing combinations obtained 0.4 and 0.39 also with the Hessian Laplace detector (0.29 recall and 0.63 precision). However, even though approximate nearest neighbors is used, each image takes around 2 seconds to be processed.

Another way to analyze the results consists in prioritizing the time and select the fastest ones. Those combinations that improved the f-measure with respect to faster combinations for those below 1 second for image have been selected as interesting. Table 5.2 shows the parameters of the chosen combinations and Table 5.3 detailed performance results.

Method	Time (sec)	Recall	Precision	F-Measure
Config 1	0.3689	0.15	0.51	0.23
Config 2	0.4206	0.14	0.87	0.24
Config 3	0.5240	0.17	0.47	0.25
Config 4	0.5471	0.17	0.9	0.28
Config 5	0.5987	0.19	0.87	0.31
Config 6	2.0335	0.28	0.64	0.39

Table 5.3: Detailed results for the chosen configurations in increasing time order.

Object	Config 1		Config 2		Config 3		Config 4		Config 5		Config 6	
	Rec	Pre	Rec	Pre	Rec	Pre	Rec	Pre	Rec	Pre	Rec	Pre
Grey battery	0	0	0	0	0	0	0	0	0	0	0	0
Red battery	0	0	0	0	0.02	0.05	0	0	0	0	0	0
Bicicle	0.54	0.52	0.52	1.00	0.33	0.52	0.36	0.89	0.38	0.90	0.33	0.62
Ponce book	0.67	0.75	0.69	0.93	0.79	0.87	0.78	0.94	0.83	0.91	0.72	0.84
Hartley book	0.58	0.93	0.58	0.93	0.86	0.77	0.88	0.88	0.95	0.85	0.81	0.73
Calendar	0.44	0.65	0.35	0.86	0.56	0.66	0.56	0.79	0.56	0.79	0.79	0.71
Chair 1	0.03	0.08	0.02	0.33	0	0	0	0	0.01	1.00	0.54	1.00
Chair 2	0	0	0	0	0	0	0	0	0	0	0	0
Chair 3	0	0	0	0	0.01	0.25	0	0	0	0	0.05	0.50
Charger	0.03	0.20	0.03	0.50	0	0	0	0	0	0	0.18	0.14
Cube 1	0.11	0.05	0.18	0.50	0.11	0.08	0.07	0.40	0.18	0.50	0.32	0.28
Cube 2	0.62	0.28	0.67	0.67	0.71	0.11	0.76	0.59	0.76	0.55	0.52	0.38
Cube 3	0.53	0.22	0.31	0.50	0.50	0.25	0.59	1.00	0.66	1.00	0.66	0.45
Extingisher	0	0	0	0	0	0	0	0	0	0	0	0
Monitor 1	0	0	0	0	0.01	0.05	0.01	1.00	0.04	0.75	0.15	0.63
Monitor 2	0	0	0	0	0	0	0	0	0	0	0	0
Monitor 3	0	0	0	0	0	0	0	0	0	0	0.02	0.33
Orbit box	0	0	0	0	0	0	0	0	0	0	0	0
Dentifrice	0	0	0	0	0	0	0	0	0	0	0	0
Poster CMPI	0.18	0.44	0.26	1.00	0.31	0.63	0.41	1.00	0.46	0.95	0.23	0.82
Phone	0	0	0	0	0	0	0	0	0	0	0	0
Poster Mystrands	0.20	0.56	0.20	0.71	0.40	0.43	0.36	0.75	0.44	0.65	0.36	0.60
Poster spices	0.38	0.77	0.42	0.94	0.54	0.79	0.53	0.87	0.58	0.87	0.56	0.92
Rack	0.26	0.59	0.26	1.00	0.10	0.80	0.10	1.00	0.23	1.00	0.77	0.79
Red cup	0	0	0	0	0	0	0	0	0	0	0.22	0.29
Stapler	0	0	0	0	0	0	0	0	0	0	0.03	0.33
Umbrella	0	0	0	0	0	0	0	0	0	0	0	0
Window	0.10	0.53	0.04	0.90	0.08	0.28	0.02	0.67	0.02	0.71	0.27	0.42
Wine bottle	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.4: Recall and precision of each object for all combinations

5.3 Evaluation of Selected Configurations

This section presents the results obtained applying the parameter combinations previously selected to all the sequences in the dataset.
In general all possible combintions of parameters performed better in well textured and flat objects, like the books or posters. For example the *Hartley book* or the *calendar* had an average recall across the six configurations (see Table 5.2 for the configuration parameters) of 0.78 and 0.54 respectively. This is not surprising as the SIFT descriptor assumes local planarity, and depth discontinuities can severely degrade descriptor similarity. On average, textured objects achieved a recall of 0.53 and a precision 0.79 across all sequences. Objects only defined by shape and color were in general harder or even impossible to detect, as can be seen in Table 5.4. Recall for this type of objects was only 0.05 on average. Configuration 6, that used the Hessian Laplace detector, exhibited a

Object	Config 1	Config 2	Config 3	Config 4	Config 5	Config 6
Normal	0.26	0.25	0.26	0.28	0.3	0.33
Blur	0.1	0.1	0.16	0.15	0.18	0.25
Occluded	0.16	0.14	0.14	0.12	0.14	0.34
Illumination	0	0	0.06	0.06	0.06	0.06
Blur+Occl	0.06	0.04	0.08	0.06	0.09	0.14
Occl+Illum	0.08	0.08	0.08	0.08	0.08	0.06
Blur+Illum	0	0	0	0	0	0

Table 5.5: Recall depending on image characteristics. *Normal* stands for object instances with good image quality and *blur* for blurred images due to motion, *illumination* indicates that the object instance is in a highlight or shadow and therefore has low contrast. Finally the last three rows indicate that the object instance suffers from two different problems at the same time.

notably better performance for some objects of this type, for example the *chair*, obtained a recall of 0.54, or the *rack* that obtained a 0.77 recall. Finally, and somewhat surprisingly, objects with a repetitive texture such as the *landmark cubes* had a quite good recall of 0.46 on average. Furthermore, the result becomes even better if we take into consideration that besides the self-similarity, all three *landmark cubes* were also similar to one another.

Regarding the image quality parameters (see Table 5.5), all combinations behaved in a similar manner: the best recall, as expected, was obtained by images not affected by blur, occlusions or strong illumination changes. From the different disturbances, what was tolerated best was occlusion, followed by blur and then by illumination. Combinations of problems also had a demolishing effect in the method performance as seen in the last three rows of Table 5.5, being the worst case the combination of *blur* and *illumination* that had 0 recall. Object instance size (for objects with a bounding box defining an area bigger than 5000 pixels) did not seem to have such an impact in performance as image quality has. However this has not yet been rigorously analyzed and is left for future work.

As predicted in Section 5.2, RANSAC and the heuristics significantly improved precision without affecting recall.

Finally, we have evaluated the exactitude in the detection of the objects by the ratio of overlap between the ground truth bounding box and the detected object instance as calculated in Equation 5.3. As can be seen in Figure 5.10, on average 70% of true positives have a ratio of overlap superior to 80% with the SIFT object recognition method regardless of the parameter combination. Furthermore, we found no appreciable advantage on overlap for any object type or instance viewing conditions, although a more in-depth analysis of this should be addressed in future work.

In order to put into context the results obtained with the selected configurations, we have also evaluated the four configurations that obtained the overall best recall and the four that obtained the overall best precision. As can be seen

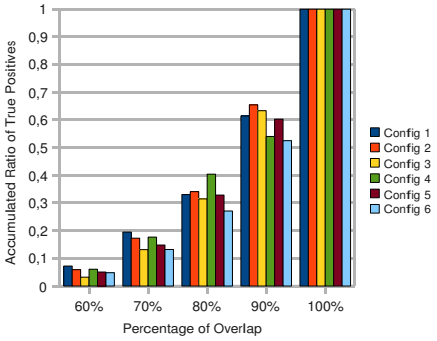


Figure 5.10: Accumulated frequencies for ratio of overlap between the ground truth bounding box and the detected bounding box for correctly found objects (true positives). An object is considered correctly detected if the ratio of overlap between the bounding boxes computed with equation 5.3 is 50% or more.

in Table 5.6, the attained recall in the selected configurations was 20% lower than the maximum obtained, independently of the type of objects. Precision is more affected by the amount of texture, and differences with respect to the top performing configurations ranged from 17% to 38%.

5.4 Discussion

Experiments show that, using the SIFT object recognition approach with the proposed modifications, it is possible to precisely detect, considering all image degradations, around 60% of well-textured object instances with a precision close to 0.9 in our challenging dataset. Even detectors known to sacrifice repeatability (probability of finding the same feature region in slightly different viewing conditions) for speed such as the SURF obtain reasonable results. Performance degrades for objects with repetitive textures or no texture at all. Regarding image disturbances, the approach resisted well occlusions, since the SIFT object recognition method is able to estimate a reliable transformation as long as the visible part of the object contains enough texture (and a minimum number of correct matches, three by default) but not so well blur due to motion or deficient illumination.

As can be seen in Table 5.3, all but one of the selected methods had a running time lower to one second, which makes them suitable for robotic applications. The step of the algorithm that takes most of the processing time is the descriptor matching, as it has a complexity of $O(N \cdot M \cdot D)$ comparisons, where N is the number of features in the new test image, M is the number of features in the training dataset and D is the dimension of the descriptor vector. Approximate matching strategies, such as the one by Muja and Lowe (2009) used in this work, are able to reduce this cost. In our experiments we experienced only a 0.01 loss in

	Best Recall		**Best Precision**		**Selected Config.**	
	mean	std	mean	std	mean	std
Repetitively textured objects						
Recall	0.65	0.09	0.16	0.01	0.46	0.05
Precision	0.02	0.01	0.75	0.15	0.43	0.24
Textured objects						
Recall	0.70	0.03	0.28	0.03	0.53	0.10
Precision	0.05	0.02	0.96	0.02	0.79	0.09
Not textured objects						
Recall	0.21	0.01	0.01	0.01	0.05	0.04
Precision	0.03	0.01	0.62	0.32	0.24	0.21

Table 5.6: Average recall and precision of the configurations that where selected for having the best values according to these two measures in the last section. Also average results among the six selected configurations are shown for comparison. Standard deviation is provided to illustrate scatter between the selected configurations. Objects are grouped in the three "level of texture" categories in the following way: the three cubes form the repetitively textured category, the two books, the calendar and the three posters form the textured category, and the rest fall into the non textured category.

the f-measure for an up to 35 times speed-up. Furthermore, an implementation tailored to performance should be able to achieve even faster rates. A drawback of the SIFT object recognition method is that it is not robust to viewpoint change. It would be interesting to evaluate how enhancing the method with 3D view clustering as described in Lowe (2001) affects the results, as it should introduce robustness to this type of transformation.

Chapter 6

Vocabulary Tree Method

The Nister and Stewenius (2006) Vocabulary Tree approach to object classification is based on the bag of words document retrieval methods, that represent the subject of a document by the frequency in which certain words appear in text. This technique has been adapted to visual object classification substituting the words with local descriptors such as SIFT computed on image features (Csurka et al, 2004; Sivic and Zisserman, 2003).
Although recently many approaches have been proposed following the *bag of words* model, we have selected this particular one because scalability to large numbers of objects in a computationally efficient way is addressed, which is a key feature in mobile robotics.
A hierarchical vocabulary tree is used instead of a linear dictionary, as it allows to code a larger number of visual features and simultaneously reduce the look-up time to logarithmic in the number of leaves. The vocabulary tree is built using hierarchical k-means clustering, where the parameter k defines the branch factor of the tree instead of the final number of clusters like in other approaches. On the negative side, using such hierarchical dictionaries causes aliasing in cluster space (see Figure 6.8.b), that can reduce the performance of the approach.
The nodes of the vocabulary tree are weighted in accordance to its discriminative power with the Term Frequency-Inverse Document Frequency (TF-IDF) scheme to improve retrieval performance. Let n_i be the number of descriptors corresponding to the codeword i found in the query image and m_i the number of descriptors corresponding to the same codeword for a given training image. If q and d are the histogram signatures of the query and database images, then the histogram bins q_i and d_i can be defined as:

$$\begin{aligned} q_i &= n_i \omega_i \\ d_i &= m_i \omega_i \end{aligned} \tag{6.1}$$

where ω_i is the weight assigned to node i. A measure based in entropy is used to define the weights:

$$\omega_i = ln(\frac{N}{N_i}), \tag{6.2}$$

where N is the number of images in the database, and N_i is the number of images in the database with at least one descriptor vector path through node i. Since signatures will be normalized before comparison, the resulting schema is the term frequency-inverse document frequency.
To compare a new query image with a database image, the following score function is used:

$$s(q,d) = \| \frac{q}{\|q\|} - \frac{d}{\|d\|} \| \tag{6.3}$$

The normalization can be in any desired norm, but the L1-norm (also known as the "Manhattan" distance) was found to perform better both by Nister and Stewenius (2006) and in our experiments. The class of the object in the query image is determined as the dominant one in the k nearest neighbors from the database.

The second speed-up proposed by Nister and Stewenius consists on using *inverted files* to organize the database of training images. In an inverted files structure each leaf node contains the ID number of the images whose signature value for this particular leaf is not zero. To take advantage of this representation, and assuming that the signatures have been previously normalized, equation 6.3 can be simplified in the following way:

$$\begin{aligned} ||q-d||_p^p &= \sum_{i=1}^{n} |q_i - d_i|^p \qquad (6.4) \\ &= \sum_{i|d_i=0} |q_i|^p + \sum_{i|q_i=0} |d_i|^p + \sum_{i|q_i\neq 0, d_i\neq 0} |q_i - d_i|^p \\ &= ||q||_p^p + ||d||_p^p + \sum_{i|q_i\neq 0, d_i\neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p) \\ &= 2 + \sum_{i|q_i\neq 0, d_i\neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p) \end{aligned}$$

with this distance formulation one can use the inverted files and, for each node, accumulate to the sum only for the training signatures that have non-zero value. If signatures are normalized using the L2 norm (i.e. the Euclidean distance), Equation 6.4 simplifies further to:

$$||q-d||_2^2 = 2 - 2 \sum_{i|q_i\neq 0, d_i\neq 0} q_i d_i \tag{6.5}$$

and since we are primarily interested in the ranking of the distances, we can simply accumulate the products and sort the results of the different images in descending order. Furthermore, as the scalar product is linear in d_i, it can be easily partitioned. Since we have used primarily the L1 normalization in this work we have still not taken advantage of this, but it is of major importance to reduce the computational cost of the algorithm and we plan to address it in future work. Figure 6.1 shows the main steps of the Nister and Stewenius (2006) algorithm.

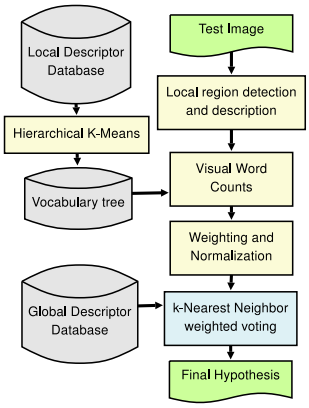


Figure 6.1: Schema of the Vocabulary Tree object recognition algorithm.

The main drawback of the Vocabulary Tree method is that it needs at least a rough segmentation of the object to be recognized. To overcome this limitation two alternatives may be used: divide the input image using a grid of fixed overlaping regions and process each region independently, or use a segmentation algorithm to yield meaningful regions to be recognized.
The first option has the advantage of simplicity and universality: Results do not depend on a particular method or set of segmentation parameters, but just on the positions and shapes of the windows evaluated. However a square or rectangular window usually does not fit correctly the shape of the object we want to detect and, in consequence, background information is introduced. Furthermore, if we want to exhaustively search the image, in the order of $O(n^4)$ overlapping windows will have to be defined, where n is the number of pixels of the image. This will we extremely time-consuming, and also fusing the classification output of the different windows into meaningful hypotheses is a non-trivial task. One way that could theoretically speed-up the sliding window process is using integral images (Viola and Jones, 2001). This strategy consists on first computing an integral image (i.e. accumulated frequencies of visual word occurrences starting from an image corner, usually top-left) for every visual word in the vocabulary tree visual word. Having the integral image precomputed for all visual words, the histogram of visual word counts for an arbitrary sub-window can be computed with four histogram operations. Let I_i be the integral image of a query image for node i of the vocabulary tree, then the histogram H of visual words counts for a given sub-window W can be computed in the following way:

$$H_i = I_i(W_{br}) + I_i(W_{tl}) - I_i(W_{tr}) - I_i(W_{bl}) \tag{6.6}$$

for all i, where W_{br}, W_{tl}, W_{tr} and W_{bl} are respectively the bottom right, top left, top right and bottom left coordinates of W.
The computational complexity of determining the visual word counts for an arbitrary sub-window is therefore $O(4 \cdot \varphi)$ operations, where δ is the size of the

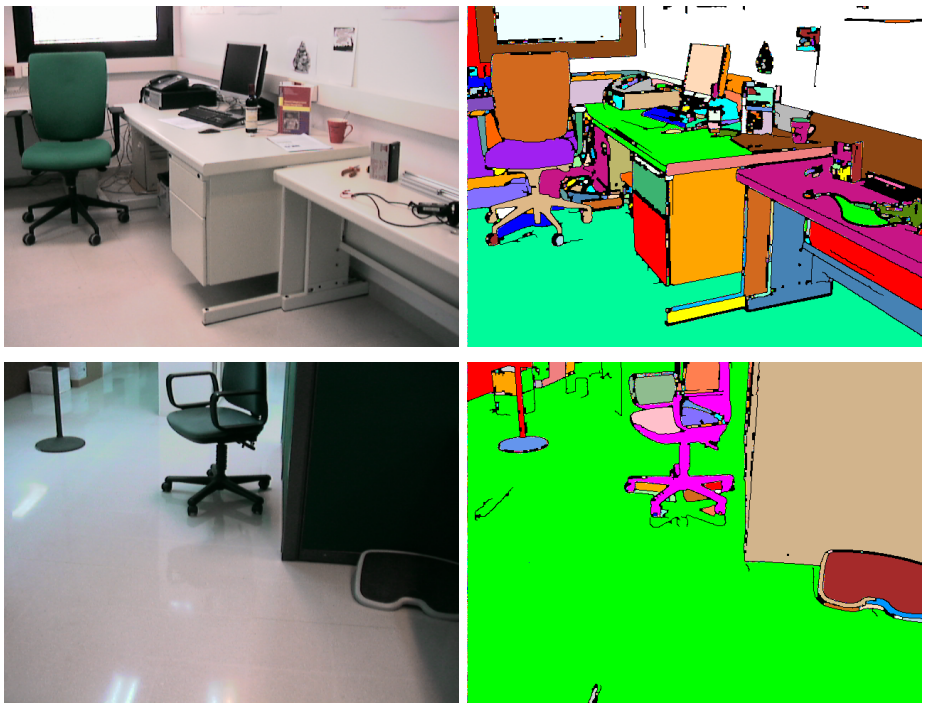


Figure 6.2: Results of the segmentation process using the *floodcanny* method. The first column shows the original images and the second column the segmented regions. Each color represents a different region, and Canny edges are superimposed for clarity.

vocabulary. Doing the same without integral images has a complexity of $O(5 \cdot \eta)$, where η is the number of visual words found in the test image. From this, it is clear that integral images are an speed up as long as φ is significantly smaller than η.

The second alternative is using a segmentation method to divide the image into a set of regions that must be recognized. Various options exist for this task which can be broadly classified as intensity based and, if stereo pairs of images are available, depth based. In this work we have evaluated an intensity based method and a depth based one. The intensity based method we propose, that we called *floodcanny*, consists on first applying the Canny edge detector to the image, and use the resulting edges as hard boundaries in a *flood filling* segmentation process. For each candidate region of an acceptable size (in our experiments, having an area bigger than 900 pixels), a set of five sub-windows of different size centered in the segmented area are defined and evaluated. In general, it is intuitive to think that, the more accurate the segmentation of the image passed to the classifier is, the better will be the results of the object recognition method. More specifically,

methods that can overcome highlights, shadows or weak reflections as the one proposed by Vazquez et al (2008) have a potential to provide more meaningful regions for the classifier, and the combination of such type of methods with appearance-based classifiers is an area of great interest, that we are willing to address in future work.
For the present work however, we have used only our proposed *floodcanny* method, which, despite of its simplicity, achieved a good segmentation results as can be seen in Figure 6.2. Furthermore, it is fast to apply (less than 30 milliseconds per image), which is very convenient given our objectives.

The second segmentation alternative proposed consisted of directly matching features between the left and right image to detect areas of constant depth. Since the geometry of the stereo cameras is known *a priori*, epipolar geometry constraints can be used together with the scale and orientation of a given feature to reduce the set of possible matches. To determine the possible location of the objects in the environment, a grid of 3D cells of different sizes is used. Reprojected features cast a vote for a cell of a grid if it lies within the 3D cell coordinates. Cells that have a minimum number of votes are reprojected to the image and added as a candidate window. It seems tempting to directly use the matched features to construct the histogram of feature word counts, as it would reduce the amount of background introduced in the visual word counts histogram. However, there is no guarantee that all features of the object have been detected in both images and matched, and the effects of missing important object features are potentially worse than introducing a small amount of background. Therefore we considered more adequate to accept all visual words close to a set of valid matches.
In future work we want to address the alternative described in the previous paragraph, and also evaluate the Vocabulary Tree method with regions generated from a dense disparity map. This latter approach would generate coherent regions of constant depth that could be used to select a dense set of image features without having to introduce background elements. Furthermore, features would only have to be computed in one image.

6.1 Databases used

In order to test and adjust the parameters of the Vocabulary Tree object recognition method, we have used two image databases in addition to the IIIA30, which we have divided in recognition and classification. These databases are detailed here:

- ASL: The ASL recognition dataset consists of nine household objects from the Autonomous Systems Lab of the ETHZ (Ramisa et al, 2008b). It consists of around 20 training images per object from several viewpoints and 36 unsegmented test images with several instances of the objects, some of them with illumination changes or partial occlusions. The train-

ing images have been taken with a standard digital camera at a resolution of 2 megapixels, while the test images have been acquired with a STH-MDCS2VAR/C stereo head by Videre design at the maximum possible resolution (1.2 megapixels). A segmented version of the training object instances has also been used in some experiments, and is referred as *segmented ASL*. Some images of the segmented version can be seen in Figure 6.1. As in the SIFT object recognition evaluation, equation 5.3 is used to determine if a object has been correctly detected.

(a)

(b)

Figure 6.3: Segmented ASL dataset images. (a) Training. (b) Testing.

- Caltech10: This is a subset of the Caltech 101 dataset[1], widely used in computer vision literature. We have taken 100 random images of the ten most populated object categories, namely: planes (lateral), bonsais, chandeliers, faces (frontal), pianos, tortoises, sails, leopards, motorbikes and clocks as seen in Figure 6.4. Training and testing subsets are determined randomly in each test. Experiments with dataset have been done following Grauman and Darrell (2005) setup: 30 random training images and the rest for testing.

6.2 Parameter Tuning

With the above described datasets we have done the following experiments in order to adjust the different parameters of the method. The experiments in this section that require no segmentation or sliding windows were done 30-fold to ensure statistical invariance when creating the tree or in the test/train data split. Similarly, unless stated otherwise in the text, the DoG feature detector

[1]The Caltech 101 dataset can be found at `http://www.vision.caltech.edu/Image_Datasets/Caltech101/`

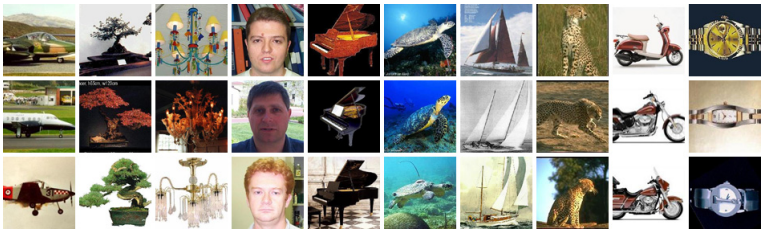


Figure 6.4: Images from Caltech10 dataset.

and SIFT descriptor were used in all experiments, and the vocabulary tree had branch factor of ten and a depth of four.

Data Type: We wanted to evaluate the impact of using the data type *unsigned char* (*uchar*) instead of *double* to represent the signature histograms, that is, 1 byte versus 8 bytes precision. For this we have tested the base method (without inverted files) with the segmented ASL and the Caltech10 datasets. As can be seen in Table 6.1, time per image –probably also thanks to a better use of the cache memory– and specially memory usage improve substantially with the unsigned char data type at the cost of some precision in the case of the segmented ASL dataset. As our aim is to obtain a fast and scalable method, we used unsigned char data type in the following experiments.

	ASL		Caltech10	
Data Type	uchar	double	uchar	double
Recall	75.48%	80.11%	54.35%	54.60%
Speed	12.39 imas/s	10.54 imas/s	7.63 imas/s	5.76 imas/s
Memory Size	1.09MB	8.74MB	1.4MB	11.23MB

Table 6.1: Results of the comparison between *uchar* i *double*

Norm: Our objective with this second experiment is to assess which normalization method achieved best performance and speed. Again we have used the segmented ASL and the Caltech10 datasets, with the same parameter settings but only with the unsigned char data type. In concordance with the observations of Nister and Stewenius (2006), the L_1 norm obtained better results in our experiments. One explanation could be the loss in precision of the more complex operations, and the discretization of the data to *unsigned char*.
In the case of the segmented ASL dataset, the difference is only 4.63% less in the case of L_2 norm. However, in the case of Caltech10 the difference increases notably, and goes from 54.35% in the case of L_1 norm to 36.43% with the L_2. Even though L_2 norm is slightly faster than L_1, it does not compensate the drop

	ASL		Caltech10	
Norm	L_1-norm	L_2-norm	L_1-norm	L_2-norm
Recall	75.48%	70.85%	54,35%	36.43%
Speed	12.39 imas/s	13.72 imas/s	7.63 imas/s	8.25 imas/s

Table 6.2: Results of the comparison between L_1-norm and L_2-norm

	ASL		Caltech10	
Inverted Files	No	Yes	No	Yes
Recall	75.48%	75.48%	54.35%	54.35%
Speed	12.39 imas/s	30.32 imas/s	7.63 imas/s	13.37 imas/s

Table 6.3: Results of the speed-up experiments with and without inverted files.

in performance. Consequently we will be using the L_1 norm in the remaining experiments.

Speed-up by Inverted Files: In this experiment we wanted to asses the processing time reduction introduced by using inverted files. Again the same datasets as in the two previous experiments will be used. In table 6.3 the results of the experiments for both databases are shown. As can be seen, speed has approximately doubled for both datasets. Furthermore, memory usage is lower with the inverted files structure.

Training Images: We have evaluated the Vocabulary Tree approach with three types of training image sets: twenty images taken from the testing sequences (i.e. of bad quality), the same training set as the SIFT Object recognition (i.e. just one training image of good quality for each category), and with twenty training images of good quality with different sizes and viewpoints. From these three training sets, the only one that obtained acceptable results has been the last one, and therefore all further experiments have been performed using it.

Detection with Sliding Windows: As explained above, the most straightforward approach to detecting objects in unsegmented images with the Vocabulary Tree method is using sliding windows. However it is time-consuming to evaluate every sub-window. As a means to accelerate this we have evaluated the *integral images* concept from Viola and Jones (2001). With the objective of adjusting the size of the windows to the objects we want to detect, we have studied the distribution of sizes of objects in the ASL and IIIA30 datasets. As can be seen in Figure 6.5, roughly 80% of the bounding box sides have under 220 pixels per side. Therefore, in order to minimize processing time, we considered a step of 20 pixels and rectangular windows with sides that ranged from 140 to 220 pixels. The total number of windows evaluated at each test image is 8625.

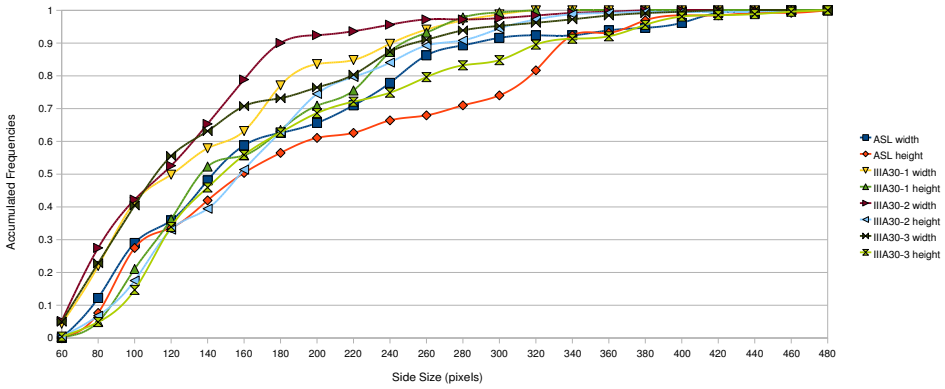


Figure 6.5: Accumulated frequencies of the ground truth bounding box side size for the objects annotated in the test sequences of the unsegmented ASL and IIIA30 datasets.

Although moving to lower values the window size span may seem attractive as more objects would fit precisely, it must be taken into account that windows are evaluated at every 20 pixels and object instances are not aligned to the sub-windows grid and, as a consequence, an offset must be expected and accommodated by the window. Furthermore the Nister and Stewenius Vocabulary tree method is able to tolerate some amount of background. For these reasons, we aimed to maximize performance with respect to number of evaluated windows with the chosen window side size span.

In our experiments, with the sliding windows approach we were able to evaluate in the order of 33 windows every second on average. However, given the high number of windows that must be evaluated, it is far from enough for real robotic applications. As discussed earlier, the speed-up provided by integral images depends on the relation between the number of features found in the image and the size of the vocabulary. However, as seen in Figure 5.4.b, Harris and Hessian based detectors find, on average, between 1500 and 2000 features for image, SIFT and SURF in the order of 500 and finally MSER only around 100 features for image.

Possible solutions to this problem have been recently proposed by different authors: Fulkerson et al (2008) suggest using agglomerative information bottleneck to reduce the size of the vocabulary tree and consequently speed up the sliding windows process, Moosmann et al (2008) use extremely randomized clustering forests of K-D trees to speed up the classification of visual words; Lampert et al (2008) disregard sliding windows and use a branch-and-bound schema over the sub-windows parameter space to direct the search to the most promising area of the image.

Figure 6.6.a shows the results obtained with the Vocabulary Tree and the sliding

windows approach in the unsegmented ASL dataset.
As seen in the figure, the number of false positives is overwhelming to say the least. To filter out as many false positives as possible, in addition to rejecting windows with less than a determined minimum number of features, we have evaluated the effect of requiring a ratio between the first and the second classes in the k-NN voting. Namely the following restriction had to be satisfied:

$$V_2 < \delta \cdot V_1 \mid \delta \in [0, 1] \tag{6.7}$$

where V_1 is the score of the most voted class, and V_2 that of the second, and δ is a threshold to reject windows without a clear winner in the k-NN voting. In Figure 6.6.b and 6.6.c results for the previous experiment applying this filtering schema with $\delta = 0.8$ and $\delta = 0.5$ are shown. As can be seen, it reduced the number of false positives in around 500 per image on average at the price of reducing also the recall by 5% in the case of setting $\delta = 0.8$ and 1500 false positives for a reduction of 11% recall when $\delta = 0.5$. In spite of this improvement, the number of false positives is still too high for the method to be usable.
In Chapter 7 we propose a method based on Reinforcement Learning that decides if a given window should be directly rejected or not by evaluating image characteristics simple to compute. This method is also able to predict which object recognition method is best (between the SIFT and the Vocabulary Tree) to recognize an hypothetic object located in the test image. This could be applied to each sub-window as a way to directly reject background without further processing.
We have also done some experiments increasing the resolution of the grid of windows in an attempt to increase recall. Figure 6.7.a shows the results of an experiment with a grid every 10 pixels instead of every 20. It must be taken into account that the change in resolution affects both the number of central points for the windows and the step of side increments. As can be seen in the table, recall improves around 5%, but also the number of false positives for image has increased by a factor of 25. In contrast with the case of 20 pixels step, applying the filtering schema proposed in the previous paragraph did not result in a significant decrease of recall, but proportionally reduced the false positives.

Detection with Segmentation: The alternative to sliding windows is using a segmentation technique to find only a few reasonable areas to search for objects. We have proposed and evaluated the *floodcanny* intensity based segmentation algorithm described earlier, and a depth based segmentation approach.
We applied the *floodcanny* to the first sequence of the IIIA30 dataset with very good results. For each region of sufficient size, a set of five windows of different sizes centered at the detected region is defined. Besides increasing recall, as can be seen in Figure 6.7.b, the number of false positives has decreased from thousands to only tens.
Despite the presented results, the segmentation scheme we have applied is not optimum, as is usually works better for large and textureless objects, that can be segmented as a big single region. Contrarily, small and textured objects pose

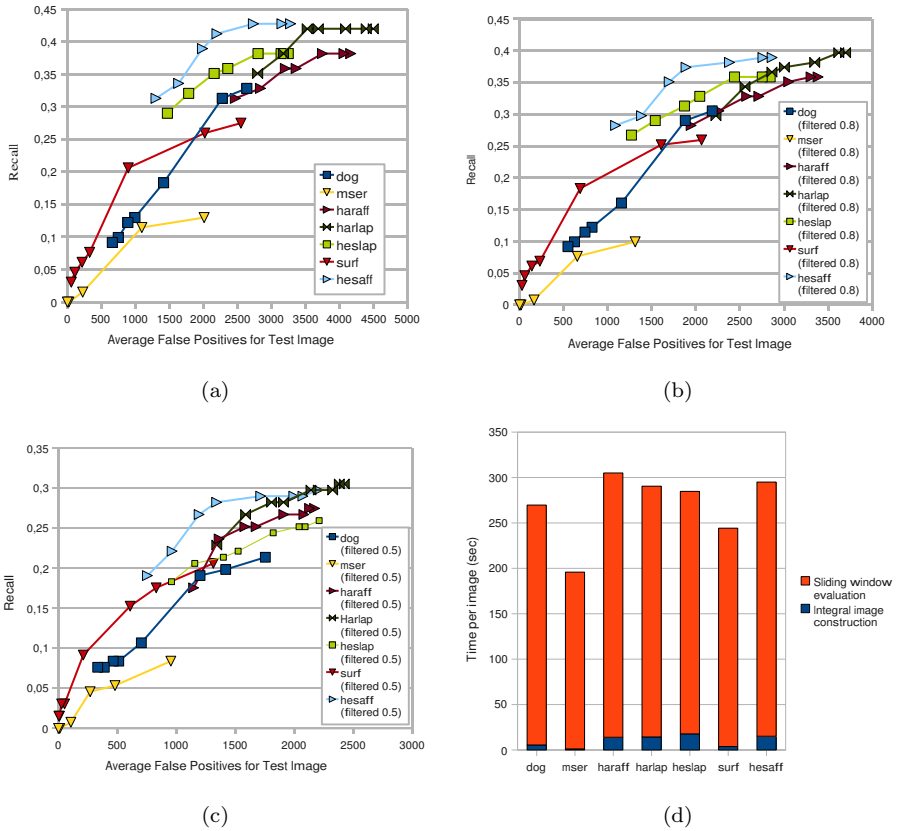


Figure 6.6: (a) Recall against average number of false positives per image found applying the Vocabulary Tree to the unsegmented ASL dataset. The other figures show results for the same experiment, but discarding windows where the second most voted class in the k-NN voting is not bigger than (b) 0.8 times and (c) 0.5 times the first. (d) Average time per image using integral images depending on feature type on the unsegmented ASL dataset.

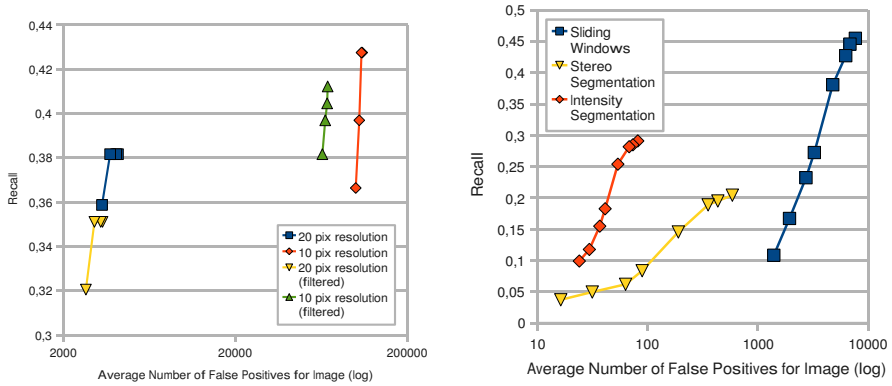
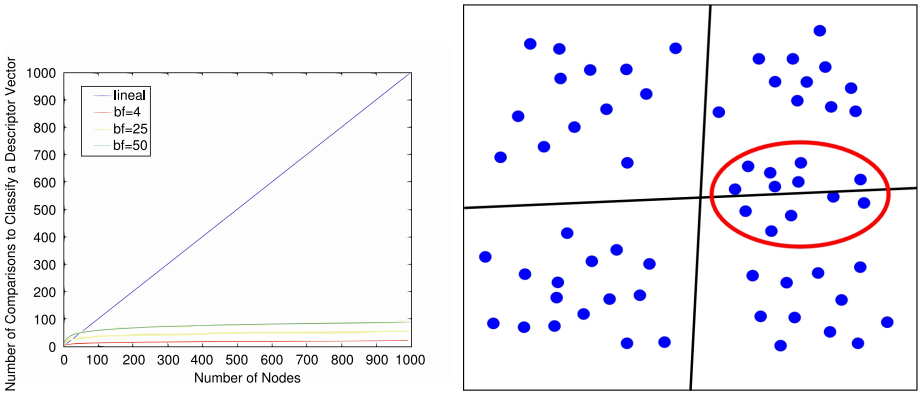


Figure 6.7: (a) Comparison of using a step of 20 pixels and a step of 10 pixels for the windows grid and sizes in the unsegmented ASL. The detector used is the Harris Affine in a tree with branch factor 10 and depth 4. (b) Results of applying Intensity Segmentation (the *floodcanny* algorithm), Stereo Segmentation and Sliding Windows to generate the sub-windows to evaluate at the first sequence of the IIIA30 dataset. For the three experiments the DoG detector and a tree with branch factor 10 and depth 4 have been used.

a problem to the *floodcanny* method, as no single large enough region can be found. Future work must include evaluating the *floodcanny* approach with more window sizes and shapes for each putative region. Also we want to evaluate the use of windows trained by the shape and scale of objects in the training set.
Regarding the depth segmentation, Figure 6.7.b also shows the results for this experiment. Although the maximum attained recall is slightly lower than that of sliding windows, it must be noted that, at a similar level of recall, false positives are much lower.

Vocabulary Tree Width and Depth: Here we evaluated different widths and depths for the vocabulary tree. Even though the computational cost of classifying a descriptor vector is much lower (see theoretical study in Figure 6.8.a), a low branch factor has the risk of introducing *aliasing* in the cluster space, as can be seen in Figure 6.8.b.
We wanted to empirically assess the effects of aliasing in our schema. Therefore, we created six vocabulary trees with the same number of leaf nodes but different branch factor and depth. Namely, the trees have branch factors: 2, 4, 8, 64, 4096 and depths: 12, 6, 4, 2, 1 respectively. Therefore we have $2^{12} = 4^6 = 8^4 = 64^2 = 4096^1$ leaf nodes.
Tables 6.4 and 6.5 show the results for the segmented ASL and Caltech 10 datasets. As can be seen, recall increases with branch factor due to a decrease of the aliasing effect. However classification time increases notably. For example,



(a) (b)

Figure 6.8: (a) Estimated cost of classifying a descriptor vector given branch factor and depth of a vocabulary tree. (b) Hierarchical clustering of the data with a low branch factor can split natural groups.

	Recall	Speed	Comparisons
V. Tree 2-12	61.44 %	29.08 imas/s	24
V. Tree 4-6	61.11 %	29.05 imas/s	24
V. Tree 8-4	63.33 %	26.47 imas/s	32
V. Tree 64-2	68.89 %	11.65 imas/s	128
V. Tree 4096-1	81.11 %	0.47 imas/s	4096

Table 6.4: Results of the vocabulary tree shape for the segmented ASL dataset.

in the segmented ASL dataset and the vocabulary tree with 8-4, recall reduces only 5% with respect to 64-2. However classification speed increases more than twice. The flat tree has no aliasing, and therefore gave the best classification results.

We have also evaluated how increasing the size of the vocabulary affects classification in unsegmented images. Figure 6.9 shows the results obtained with trees of different sizes and shapes. As found by Nister and Stewenius (2006), usually larger dictionaries help increase recall, although this only holds as long as enough training features are available. In our experiments we obtained the highest recall with a dictionary of branch factor 9 and depth 4. This is dependent on the amount of features found by the detector and the redundancy between them, but in the case presented in the figure it represents approximately a ratio of one leaf node for every 2 features.

We also found that using large dictionaries makes it more difficult to effectively model the background, and therefore more false positives were found with them. A possible solution for the background modeling problem could be, for exam-

	Recall	Speed	Comparisons
V. Tree 2-12	56.57 %	13.03 imas/s	24
V. Tree 4-6	54.29 %	14.33 imas/s	24
V. Tree 8-4	52.29 %	13.56 imas/s	32
V. Tree 64-2	58.14 %	5.24 imas/s	128
V. Tree 4096-1	60.29 %	0.05 imas/s	4096

Table 6.5: Results of the vocabulary tree shape for the Caltech10 dataset.

ple, adding samples to the background class from sub-windows that contain no object, either coming from ground truth data or on-line, in a supervised way or automatically when a confusing hypothesis is finally rejected with high confidence (e.g. by moving the robot closer to the object).
An early rejection method for unpromising windows as the one proposed in Chapter 7 has the potential to keep the improvement in recall while discarding windows that could later generate false positives because of poor background category modeling.

Number of Nearest Neighbors: In order to find a good k value for the k-NN classifier, here we experimentally evaluated the classification performance with respect to this parameter.
When applying the Vocabulary tree method to the segmented datasets, and in accordance with Duda et al (2001), the first nearest neighbor gave a good classification rate. This was especially true in the case of recognition with few training images, as in the segmented ASL dataset. We have also evaluated the effect of weighting the votes by its distance to the most voted. This helps reduce the effect of distant wrong neighbors when k is increased. Results of these experiments can be seen in Figure 6.10 for the segmented ASL dataset, and in 6.11 for the Calthech 10. In general, weighting the votes slightly decreased the recall but had more stable results as the number of nearest neighbors considered was increased.
Regarding the recognition in unsegmented images, the choice of the first nearest neighbor yielded the best recall for the filtered results as well, but also a very low precision. Overall, four or five nearest neighbors had the best balance between recall and false positives.

Feature Detectors: We have tested the seven feature detectors (i.e. Harris Affine, Hessian Affine, Harris Laplace, Hessian Laplace, MSER, SURF) using sliding windows, in the unsegmented ASL dataset. In Figure 6.6.a the recall against average number of false positives per image can be seen when varying the minimum number of features to accept a window. As in the case of the SIFT object recognition, Hessian based detectors achieved the highest recall. In this case, however, Harris detectors achieved a comparable recall for a much higher number of false positives (around 1000 more for test image). One possible expla-

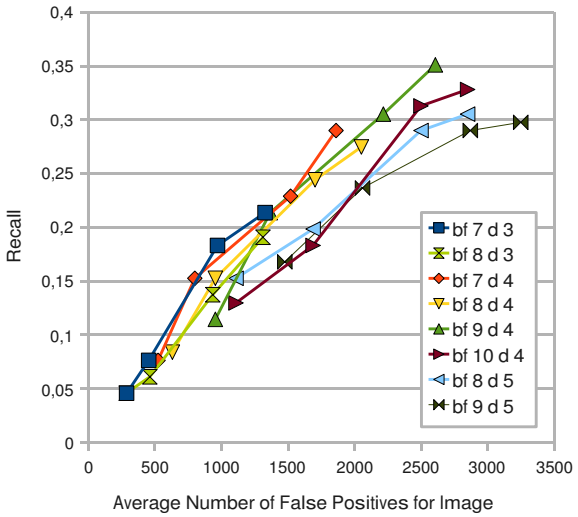
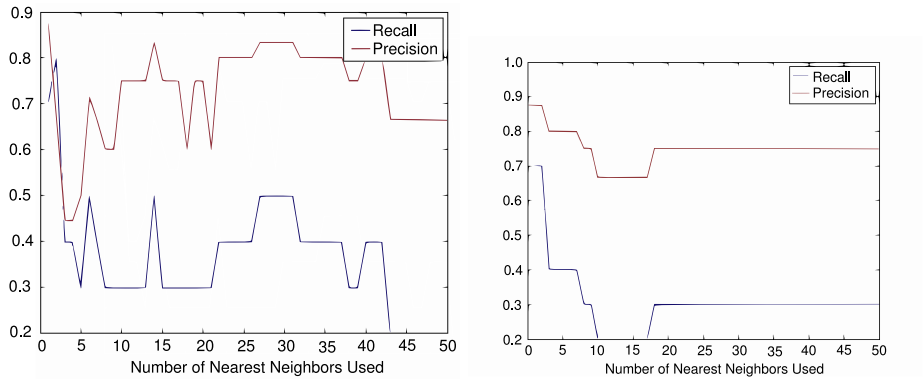


Figure 6.9: Recall against average number of false positives per image in the unsegmented ASL dataset with trees of different sizes and shapes. The detector used has been the DoG. The shape of the tree is indicated in the legend, where *bf* stands for branch factor and *d* for depth.

nation for this difference could be that corners are closer to depth discontinuities more often than blobs. Depth discontinuities violate the planarity assumption of the feature detector and introduce variations to the descriptor, which in turn will increase the probability of a failed match or a classification in an erroneous visual word. Aiming to minimize the false positives, the best results were obtained by the SURF detector (also Hessian based). However, for higher recall levels, it performed worse than all the other detectors except MSER.
Regarding computational time, differences between each detector are due to more areas skipped because of insufficient features as, once the integral image is constructed, it takes a constant time to evaluate every sub-window. In fact, various authors argue that with integral images it is more convenient to use dense features, as recall is improved with such high numbers of features while computational time is not affected using integral images (Fulkerson et al, 2008; Nowak et al, 2006).

Manually segmented images from IIIA30 dataset: As a final tuning experiment, we wanted to evaluate the achievable performance with the IIIA30 dataset if we omit the detection step. Therefore we taken one hundred segmented images (twenty for object) of five objects from the sequences dataset: *Ponce book*, *charger*, *orbit box*, *Spices poster* and *stapler*. Also six standard digital camera quality training images were taken. We called it the segmented IIIA5 dataset,



(a) (b)

Figure 6.10: (a) Results of unweighted votes for the segmented ASL dataset. (b) Results of votes weighted by distance for the segmented ASL dataset.

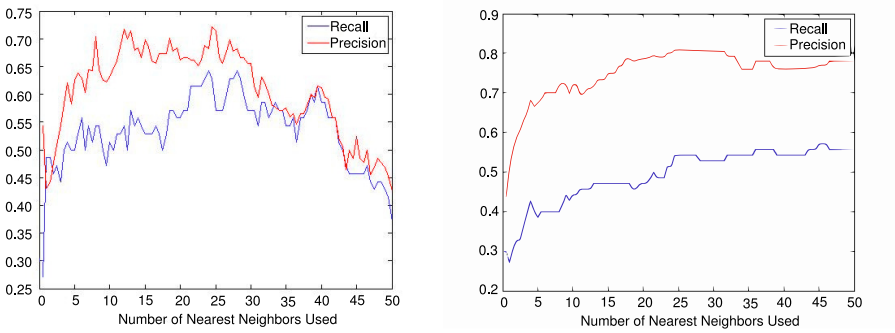
and Figure 6.13 shows some train and test images. Results of the experiment with the DoG detector and a tree with branch factor 10 and depth 4 are shown in Table 6.6.

Object	Recall	Precision
Ponce book	50,00%	71,43%
Charger	30,00%	50,00%
Orbit box	100,00%	47,62%
Poster spices	100,00%	90,91%
Stapler	30,00%	60,00%
Average	**63,99%**	**62,00%**

Table 6.6: Recall and precision for the segmented IIIA5 dataset.

6.2.1 Discussion and Selected Configuration

Except for recall, which is better for the Vocabulary Tree method, the SIFT object recognition has better results in all other aspects related to robotics.
Regarding the different alternatives considered in the experiments, the use of inverted files is the one that most clearly improves the method without negatively affecting it in any way. The use of integral images is only sensible in the case of small dictionaries and high number of features, otherwise it becomes more computationally expensive. In addition, it must be also taken into account the time employed in building the integral images structure. Contrarily, we found that large dictionaries improved recall. Experiments done considering larger window spans than the one proposed in this work did not show much



(a) (b)

Figure 6.11: (a) Results of unweighted votes for Caltech 10. (b) Results of votes weighted by distance for Caltech 10.

improvement, but this should be better analyzed in future work. Increasing the sampling resolution to construct the integral image (i.e. the jump in pixels from one bin of the integral image to the next) from every 20 pixels to every 10 pixels did increase the recall, but at the cost of 25 times more false positives.

To get rid of false positives we evaluated a filtering schema to reject windows with dubious classification results. This approach did indeed reduce around 500 to 1500 false positives for image, which is approximately 25% of total false positives generated at the cost of 5% to 11% drop in recall. However, in order to be practically usable, much more false positives should be rejected without significantly affecting the recall.

Among the feature detectors used, best results were obtained by the Hessian based ones, while the Harris based detectors attained similar recall levels but at the cost of more false positives.

We have, therefore, selected for the comparison the sub-windows generated by the *floodcanny* segmentation technique, given that a sliding windows approach with a large vocabulary tree is too computationally expensive for a robotics scenario, and the Hessian Affine feature detector and a tree with branch factor nine and depth four. This represents a ratio of approximately 12 features per leaf node of the tree. Although this ratio is higher than the one found for the DoG detector in the size and shape of the tree tuning experiment, our intuition is that the used implementation of the Hessian Affine usually finds more redundant regions than the DoG. We did indeed test the same experiment with a vocabulary tree of branch factor ten and depth four and performance was slightly worse. Results of this comparison can be seen in Figure 6.14.

It would have been interesting to consider also the depth segmentation, but unfortunately stereo pairs of images were only available for one sequence of the IIIA30 dataset.

It must be noted that, although the results obtained may seem not very good

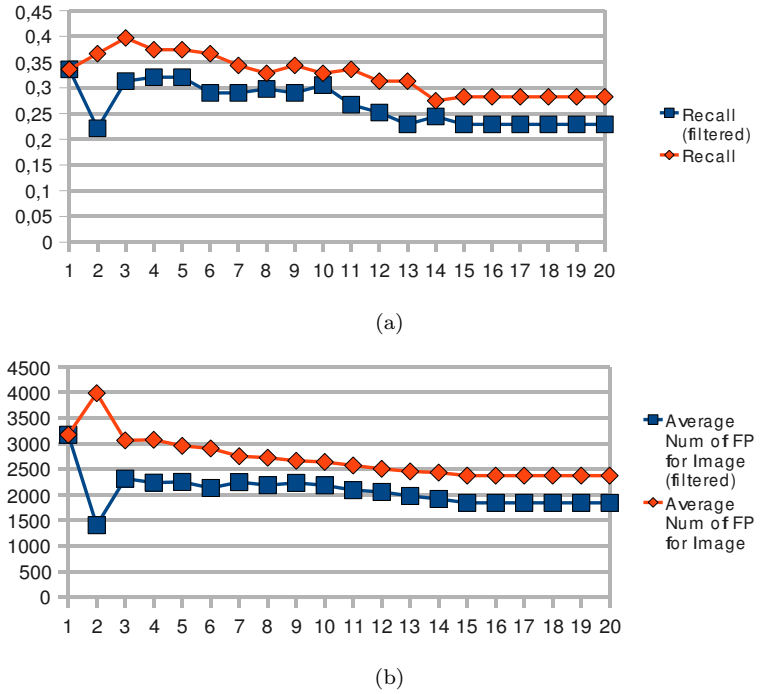


Figure 6.12: Results of different numbers of nearest neighbor choices for the unsegmented ASL dataset. (a) Recall (b) Average number of false positives per image

because of the high number of false positives, they are much better than random. In the segmented ASL dataset, each sub-window represents a 10 class classification problem, with a 0.10 probability of randomly picking the right answer. As 8625 sub-windows are evaluated at every image, the number of expected correct random classifications per image is of 862.5 (including true negatives) and, therefore, the number of expected false positives is 7762.5. However, according to the result shown in Figure 6.6.c, only approximately 1200 false positives per image were found in the sliding windows experiment. In the case of the IIIA30 dataset, the difference in the number of false positives of the method and random selection is worse, probably because of poor background modeling.

In addition, it has to be taken into account that Equation 5.3 does penalize a sliding windows based approach, as it does not enforce finding the right bounding box for the object. Therefore, such sliding windows that do not fit the object precisely enough will be discarded regardless of correctly selecting the class.

Figure 6.13: Training (up) and testing (down) images from the segmented IIIA5 dataset.

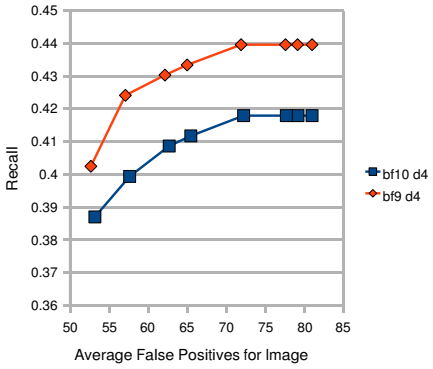


Figure 6.14: Comparison between a vocabulary tree with branch factor 10 and depth 4 and another with branch factor 9 and depth 4. The feature detector is the Hessian Affine and the test sequence is the IIIA30-1.

6.3 Evaluation of Selected Configurations

As can be seen in Table 6.3, with the segmentation schema adopted in this final experiment, we have obtained a recall better than with the SIFT method for untextured objects. Unfortunately small and textured objects are harder to detect with the current segmentation schema, as they usually do not generate a large enough uniform region. However this is not a weakness of the Vocabulary Tree method but of the segmentation approach, if it were improved, the classification results would possibly meliorate.

Objects like the computer monitors, the chairs or the umbrella had a recall comparable to that of textured objects. As can be seen in Table 6.8, a similar recall was obtained for the objects of types textured and not textured. A slightly worse recall was obtained for the repetitively textured objects, but we believe it is mostly because of the segmentation method.

Regarding the image quality parameters (see Table 6.9), the occluded objects

Objects	10nn		10nn with filtering $\delta = 0.8$		5nn		1nn		10nn with relaxed overlap	
	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec
Grey battery	0.36	0.01	0.32	0.02	0.32	0.01	0.36	0.01	0.60	0.02
Red battery	0.13	0.01	0.03	0	0.18	0.01	0.17	0.01	0.15	0.01
Bicicle	0.67	0	0.59	0	0.58	0.01	0.49	0.01	0.70	0
Ponce book	0.38	0.01	0.38	0.02	0.41	0.01	0.41	0.01	0.69	0.02
Hartley book	0.21	0	0.21	0	0.19	0	0.21	0	0.81	0.01
Calendar	0.18	0	0.09	0	0.15	0	0.12	0	0.53	0.01
Chair 1	0.70	0.05	0.69	0.06	0.72	0.05	0.78	0.06	0.71	0.06
Chair 2	0.14	0.04	0.09	0.04	0.15	0.05	0.11	0.03	0.19	0.06
Chair 3	0.01	0.04	0	0	0.04	0.10	0.02	0.05	0.01	0.04
Charger	0.11	0	0	0	0	0	0	0	0.11	0
Cube 1	0.36	0	0.36	0	0.5	0	0.43	0.01	0.43	0
Cube 2	0.11	0	0.11	0	0.11	0	0.17	0	0.28	0.01
Cube 3	0.06	0	0.06	0	0.03	0	0.09	0	0.15	0.01
Extingisher	0	0	0	0	0	0	0	0	0	0
Monitor 1	0.13	0.01	0.12	0.02	0.10	0.01	0.15	0.02	0.22	0.02
Monitor 2	0.45	0.11	0.42	0.13	0.51	0.11	0.57	0.08	0.58	0.15
Monitor 3	0.77	0.16	0.77	0.17	0.66	0.14	0.71	0.09	0.93	0.21
Orbit box	0.14	0	0.14	0	0.14	0	0.14	0	0.71	0
Dentifrice	0	0	0	0	0	0	0.13	0	0	0
Poster CMPI	0.26	0.02	0.23	0.02	0.26	0.03	0.26	0.02	0.26	0.02
Phone	0.06	0.01	0.06	0.01	0.03	0	0.04	0	0.06	0.01
Poster Mystrands	0.28	0.03	0.28	0.03	0.24	0.04	0.24	0.03	0.28	0.03
Poster spices	0.46	0.02	0.46	0.02	0.35	0.02	0.46	0.03	0.59	0.03
Rack	0.60	0.06	0.58	0.07	0.60	0.07	0.58	0.06	0.82	0.09
Red cup	0	0	0	0	0	0	0	0	0	0
Stapler	0.18	0.02	0.18	0.02	0.13	0.02	0.24	0.01	0.21	0.02
Umbrella	0.02	0.01	0.02	0.01	0.01	0	0	0	0.02	0.01
Window	1.00	0.07	1.00	0.07	1.00	0.07	1.00	0.07	1.00	0.07
Wine bottle	0	0	0	0	0	0	0	0	0.40	0.01

Table 6.7: Precision and recall for all the objects of the IIIA30 dataset in the final Vocabulary Tree experiment (i.e. tree with branch factor 9 and depth 4, and features found with the Hessian Affine detector). Different choices of parameters for the classifier are displayed. Also, the last column, shows the results obtained using Equation 5.4 instead of Equation 5.3 to measure overlap.

obtained a higher recall level, but this was because, as mentioned in the previous discussion, the sliding windows approach taken in this experiment does not enforce a precise detection and, therefore, Equation 5.3 discards hypotheses correctly detecting object instances. When Equation 5.4 was used for all objects, instead of restricting it only to the occluded ones, recall for objects with *normal* and *blurred* viewing conditions is increased. Also, as can be seen in Figure 6.15, the percentage of detected objects with a degree of overlap from 90% to 100% between the found and the ground truth bounding box was increased by 14%, showing that, although not precisely, the considered windows did overlap almost the whole object region.

6.4 Discussion

With the selected configuration we obtained an average recall of 30%. More importantly, this approach has been able to detect objects that the SIFT could not find because of its restrictive matching stage. However, also 60 false positives per image on average were detected with the selected configuration, which represents a precision of 2% on average.
In the light of the performed experiments, it seems clear that the Vocabulary Tree method cannot be directly applied to a mobile robotics scenario, but some

	10nn	10nn-0.8	5nn	1nn	10nn-relaxed
Repetitively textured objects					
Recall	0.18	0.18	0.21	0.23	0.29
Prec	0	0	0	0	0.01
Textured objects					
Recall	0.29	0.27	0.26	0.28	0.53
Prec	0.02	0.02	0.02	0.02	0.02
Not textured objects					
Recall	0.29	0.26	0.27	0.29	0.39
Prec	0.03	0.03	0.03	0.03	0.04

Table 6.8: Precision and recall depending on texture level of the objects in the final experiment with the Nister and Stewenius (2006) Vocabulary Tree. The objects are grouped in the same way as in Table 5.6. The title *10nn-0.8* stands for 10nn with filtering $\delta = 0.8$, and *10nn-relaxed* for 10nn with relaxed overlap.

strategy to drastically reduce the number of false positives is necessary. In addition to reduce false positives to acceptable levels, it will be necessary to accelerate a bit the detection step in order to process images coming from the robot cameras at an acceptable rate. Improving the segmentation strategy, or using a technique such as the one presented in Chapter 7 will surely help improve the accuracy.

Nevertheless, we found that the Vocabulary Tree method was able to detect objects that were inevitably missed by the SIFT Object Recognition method. Furthermore, as it is a hot research topic, new and promising *bag of features* type approaches are currently being proposed, such as the aforementioned Fulkerson et al (2008) approach, the one by Moosmann et al (2008) and specially the one by Lampert et al (2008). Although we would have liked to evaluate all these new strategies here as well, time constraints make it impossible to address it now and we must leave it for future work.

	10nn	10nn-0.8	5nn	1nn	10nn-relaxed
Normal	0.24	0.23	0.24	0.25	0.45
Blur	0.29	0.28	0.28	0.3	0.46
Occluded	0.64	0.61	0.62	0.62	0.64
Illumination	0.06	0.06	0.06	0.11	0.11
Blur+Occl	0.43	0.41	0.43	0.46	0.43
Occl+Illum	0.11	0.11	0.08	0.08	0.11
Blur+Illum	0.14	0	0	0	0.14

Table 6.9: Recall depending on image characteristics. *Normal* stands for object instances with good image quality and *blur* for blurred images due to motion, *illumination* indicates that the object instance is in a highlight or shadow and therefore has low contrast. Finally the last three rows indicate that the object instance suffers from two different problems at the same time.

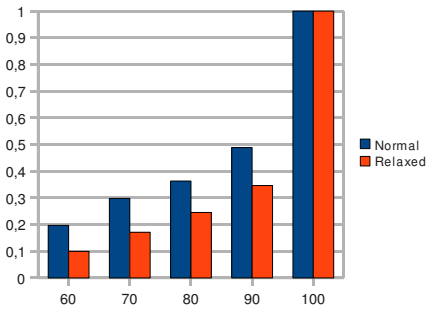


Figure 6.15: Accumulated frequencies for the ratio of overlap between the ground truth bounding box and the detected bounding box for correctly found objects (true positives). For the "Normal" category (corresponds to the 10nn of previous results) Equation 5.3 is used to determine the ratio, while for the "Relaxed" category, Equation 5.4 is used. In both cases an object is only accepted if the ratio found is higher than 50%.

Chapter 7

Object Recognition Method Selection with Reinforcement Learning

7.1 Introduction

After reviewing the two chosen object recognition methods with their advantages and drawbacks in a mobile robotics scenario, with the variety of conditions that they will face, one can conclude that choosing, a priori, which object recognition method a robot should have, is not the best design option. In this kind of application, the robot should be able to decide by itself which object recognition method should be used, depending on the current conditions of the world.
In this chapter we propose the use of Reinforcement Learning to decide on line which method should be used to identify objects in an image, aiming also to minimize computing time. To evaluate this idea we implemented a system that is able to choose between the two object recognition algorithms used in this work based on simple attributes extracted on-line from the images, such as mean intensity and intensity deviation. In addition, it is also capable of deciding that an image is not suitable for analysis, and thus discard it.

7.2 Reinforcement Learning and its applications in Computer Vision

Reinforcement Learning (Sutton and Barto, 1998) is concerned with the problem of learning from interaction to achieve a goal, for example, an autonomous agent interacting with its environment via perception and action. On each interaction step the agent senses the current state s of the environment, and chooses an action a to perform. The action a alters the state s of the environment, and a

scalar reinforcement signal r (a reward or penalty) is provided to the agent to indicate the desirability of the resulting state. In this way, "The RL problem is meant to be a straightforward framing of the problem of learning from interaction to achieve a goal" (Sutton and Barto, 1998).
Formally, the RL problem can be formulated as a discrete time, finite state, finite action Markov Decision Process (MDP) (Mitchell, 1997). Given:

- finite set of states $s \in \mathcal{S}$ that the agent can achieve;
- A finite set of possible actions $a \in \mathcal{A}$ that the agent can perform;
- A state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$, where $\Pi(\mathcal{S})$ is a probability distribution over $\mathcal{S}$;
- A finite set of bounded reinforcements (payoffs) $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$,

the task of a RL agent is to find out a stationary policy of actions $\pi^* : \mathcal{S} \rightarrow \mathcal{A}$ that maps the current state s into an optimal action(s) a to be performed in s, maximizing the expected long term sum of values of the reinforcement signal, from any starting state.
The policy π is some function that tells the agent which actions should be chosen, and is learned through trial-and-error interactions of the agent with its environment. Several algorithms were proposed as a strategy to learn an optimal policy π^* when the model ($\mathcal{T}$ and $\mathcal{R}$) is not known in advance, for example, the Q–learning (Watkins, 1989) and the SARSA (Rummery and Niranjan, 1994) algorithms.
The Q–learning algorithm was proposed by Watkins (1989) as a strategy to learn an optimal policy π^* when the model ($\mathcal{T}$ and $\mathcal{R}$) is not known in advance. Let $Q^*(s, a)$ be the reward received upon performing action a in state s, plus the discounted value of following the optimal policy thereafter:

$$Q^*(s, a) \equiv R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V^*(s'). \tag{7.1}$$

The optimal policy π^* is $\pi^* \equiv \arg\max_a Q^*(s, a)$. Rewriting $Q^*(s, a)$ in a recursive form:

$$Q^*(s, a) \equiv R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \max_{a'} Q^*(s', a'). \tag{7.2}$$

Let $\hat{Q}$ be the learner's estimate of $Q^*(s, a)$. The Q–learning algorithm iteratively approximates Q^*, i.e., the $\hat{Q}$ values will converge with probability 1 to Q^*, provided the system can be modeled as a MDP, the reward function is bounded $(\exists c \in \mathcal{R}; (\forall s, a), |R(s, a)| < c)$, and actions are chosen so that every state-action pair is visited an infinite number of times. The Q learning update rule is:

$$\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha \left[r + \gamma \max_{a'} \hat{Q}(s', a') - \hat{Q}(s, a) \right], \tag{7.3}$$

where s is the current state; a is the action performed in s; r is the reward received; s' is the new state; γ is the discount factor ($0 \leq \gamma < 1$); $\alpha = 1/(1 + visits(s, a))$, where $visits(s, a)$ is the total number of times this state-action pair has been visited up to, and including, the current iteration.

Several researchers have been using RL as a technique to optimize active vision, image segmentation and object recognition algorithms. The area of Computer Vision on which RL was first applied was in Active Vision. Whitehead and Ballard (1991) proposed an adaptive control architecture to integrate active sensorymotor systems with RL based decision systems. Although the work is theoretical and did not make use of real sensors, they were able to describe a system that learns to focus its attention on the relevant aspects of the domain as well as control its behavior, in a simple block manipulation task. Several researchers have been applying RL to active vision since then, for example Minut and Mahadevan (2001) have applied RL for visual attention control, proposing a model of selective attention for visual search tasks, such as deciding where to fixate next in order to reach the region where an object is most likely to be found. Darrell and Pentland (1996a,b) also address visual attention problem: they proposed a gesture recognition system that guides an active camera to foveate salient features based on a Reinforcement Learning paradigm. An attention module selects targets to foveate based on the goal of successful recognition, learning where to foveate to maximally discriminate a particular gesture. Finally, in Darrell (1998) is shown how a concise representation of active recognition behavior can be derived from hidden-state Reinforcement Learning techniques.

Paletta and Pinz (2000) have applied RL in an active object recognition system, to learn how to move the camera to informative viewpoints, defining the recognition process as a sequential decision problem with the objective of disambiguating initial object hypotheses. For these authors, "Reinforcement Learning provides then an efficient method to autonomously develop near-optimal decision strategies in terms of sensorimotor mappings" (Paletta et al, 1998). Borotschnig et al (1999) continued in the same line of work, building a system that learns to reposition the camera to capture additional views to improve the image classification result obtained from a single view. More recently, Paletta et al (2005) proposed the use of Q-learning to associate shift of attention actions to cumulative reward with respect to object recognition. In this way, the agent learns sequences of shifts of attentions that lead to scan paths that are highly discriminative with respect to object recognition.

Less work has been done on the use of RL for image segmentation and object recognition. Peng and Bhanu (1998a) used RL to learn, from input images, to adapt the image segmentation parameters of a specific algorithm to the changing environmental conditions, in a closed-loop manner. In this case, the RL creates a mapping from input images to corresponding segmentation parameters. This contrasts with great part of the current computer vision systems whose methodology is open-loop, using image segmentation followed by object recognition algorithms. Peng and Bhanu (1998b) improve the recognition results over time by using the output at the highest level as feedback for the learning system, and

has been used to learn the parameters of image segmentation and feature extraction and thereby recognizing 2-D objects, systematically controlling feedback in a multilevel vision system. The same authors presented a general approach to image segmentation and object recognition that can adapt the image segmentation algorithm parameters to the changing environmental conditions, in which segmentation parameters are represented by a team of generalized stochastic learning automata and learned using connectionist Reinforcement Learning techniques. Results were presented for both indoor and outdoor color images, showing a performance improvement over time for both image segmentation and object recognition using RL (Bhanu and Peng, 2000).

Taylor (2004) also followed this line of research, applying RL algorithms to learn parameters of an existing image segmentation algorithm. Using the Fuzzy ARTMAP artificial neural network, he was able to optimize ten parameters of Wolf and Jolion (2003) algorithm for text detection in still images. The parameters learned by RL were shown to be superior to the parameters previously recommended. Other applications of RL to learn parameters of image segmentation algorithms include: contrast adaptation (Tizhoosh and Taylor, 2006), finding the appropriate threshold in order to convert an image to a binary one (Yin, 2002; Shokri and Tizhoosh, 2003, 2004, 2008; Sahba et al, 2008) and detection of patterns in satellite images (Hossain et al, 1999).

Finally, Draper et al (1999) modeled the object recognition problem as a Markov Decision Problem, and proposed a theoretically sound method for constructing object recognition strategies by combining Computer Vision algorithms to perform segmentation. The authors tested their method in a real system, learning sequences of image processing operators for detecting houses in aerial images.

In summary, Reinforcement Learning has been widely used in the Computer Vision field in particular cases, mainly: to optimize the performance of active vision systems; to decide where the focus of attention should be in order to accomplish a certain task; to learn how to move a camera to more informative viewpoints and to optimize parameters of existing and new computer vision algorithms, such as thresholds, contrast and internal parameters. In these cases, the resulting systems and algorithms have been very successful ones.

RL has also been used for constructing object segmentation and recognition strategies by combining computer vision algorithms. However, results in this area have not been as good as those of RL applied to active vision or parameter optimization: it usually has been limited to a very specific image domain, such as the one in Draper et al (1999).

The main limitations which arise when using Reinforcement Learning are, first, that the reward value associated with a situation is usually not directly available, and thus rewards are results of indirect definitions. RL requires that a certain amount of knowledge about the world is available in the form of a training set, which is not the case in many vision tasks. Second, the large state space difficults convergence of RL algorithms raises performance issues, as the learning phase can take a long time.

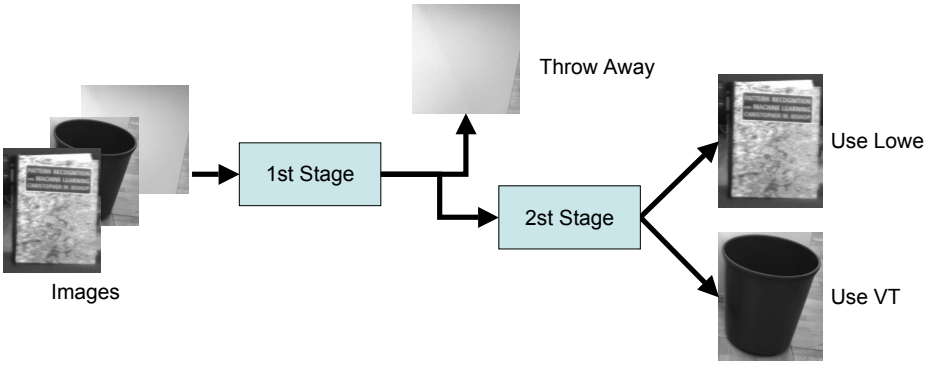


Figure 7.1: The two stage decision problem.

7.3 Learning to Select Object Recognition Methods

In order to decide which algorithm should be used by the learning agent, the RL problem was defined as a 2 stage decision problem, with 2 possible actions in each stage: In the first one, the agent must decide if the image contains an object, and thus must be recognized, or if the image does not contain objects, and can be discarded, saving processing time. In the second stage, the agent must decide which object recognition algorithm should be used: Lowe's SIFT Object Recognition method or the Nister and Stewenius Vocabulary Tree (VT) algorithm (see Figure 7.1).

To learn how to select the object recognition method appropriate for one image at one stage we propose to use Reinforcement Learning as a classification method. In this approach the state space is defined as a combination attributes extracted from the images plus the possible classification of the image. For example, for the first stage, the state can be defined as a combination of mean image intensity and standard deviation and a value defining if the image is a background and can be discarded or if contains objects.

We also define a new type of action, called "update action". Update actions are not real actions happening in the world, but actions that update the value of a state-action pair $Q(s, a)$ at one state using the value of a neighbor pair. For example, if the state space is composed of image intensity and standard deviation, the $Q(s, a)$ table would be represented as two dimensional matrix containing the possible values of intensity (0 to 255) and standard deviation (0 to 255), and update actions are done between one state and his 4-neighbours located above, below, at left and at right. In a 3-dimensional state space, update actions can be made using 8-neighbours (two in each direction of the space) and so on.

The rewards used during the learning phase are computed using a set of training images. If, during the exploration, the learning agent reaches a state where

a training image exists and the state corresponds to the correct classification of the image, the agent receives a reward. Otherwise the reward is zero. For example, in the first stage of the decision, if we have a training image that does not contain an object, with mean intensity value of 50 and standard deviation of 10, a reward is given when the agent moves to the state (mean = 50, std = 10, classification = discard). An interesting propriety of this approach is that the rewards can be pre-computed, creating a reinforcement table that can be used during the learning phase.
Formally, a MDP can be defined for each stage as:

- The set of update actions $a \in \mathcal{A}$ that the agent can perform, defined as update the Q value using the value of a neighbor.
- The finite set of states $s \in \mathcal{S}$ in this case is the n-dimensional space of values of the attributes extracted from the images plus its classification;
- The state transition function allows updates to be made between any pair of neighbors in the set of states.
- The reinforcements $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ are defined using a set of training images.

In this approach, a RL method is used as a classifier, and must have two processing phases: the training phase, where Reinforcement Learning is performed over a set of pre-classified images, i.e., images to which we know what the best algorithm to use is, and the execution phase, where the results from the learning are used to decide which algorithms to apply to other images.
During the training phase, learning an optimal policy to solve the MDP means to learn a mapping from images (or more specifically image attributes) to image classes (or algorithms classes). Although several RL algorithms can be used to do this, the RL algorithm used in this implementation is the Q-learning (Watkins, 1989), because it directly approximates the optimal policy independently of the policy being followed (it is an off-policy method), allowing the state and the action to be executed by the agent to be selected randomly. Using the Q-Learning, at each stage the agent chooses a system state s. Then, it selects an update action to be executed, computes the reward and updates the value function.
The learning algorithm used is as follows:

```
Initialize Q(s,a).
Choose a start state s, randomly.
do {
   Choose an action, randomly.
   Execute a, observe s', compute the reward.
   Update the Q value.
   s = s'.
} until the Q values converge.
```

0000000000	1111111110
0000010000	1111111111
2000011000	2011111111
0000001100	2221111111
0200101000	2222111111
0020200110	2222211111
0000010000	2222111111
0200011000	2220111111
2021000000	2221111111

Figure 7.2: Example of how the classification works: the reinforcements used (left) and the resulting classification table (right).

To better understand what is happening during the learning phase, we can compare our approach to a robot moving in a two-dimensional grid. Every time the robot finds a "goal" state and receives a reward, the state-action pair where the robot was before reaching the goal state is updated. Every time the robot moves, it iteratively updates the origin state-action pair. By doing this a large number of times, the reward is spread over the Q-table, and a robot will know what to do to reach the goal state (will have learned the optimal policy).
In the learning phase, every time the "robot" reaches a state where there is an image from the training set, it receives a reward, and the state-action pair where the "robot" was before is updated. Every time a new state action pair is randomly chosen, it is iteratively updated. By doing this a large number of times, the reward is spread over the Q-table, and every state-action pair will contain information about what to do with an image with those characteristics (will have learned a mapping from image to actions).
Figure 7.2 shows an example of the use of RL as an image classification method: in each figure the rows and columns represents two values of the attributes extracted from the images (for example, mean image intensity are rows and standard deviation of the image intensity are the columns), the value represents what is the classification of an image, where zero means that there is not an example with that characteristics, and "1" and "2" are two possible classes.
The table of the left in figure 7.2 presents the reinforcement table, created before the training phase using the training images and their classification, which consists of what is the algorithm that was able to classify them. The table on the right in the same figure shows the results of applying the RL algorithm during the learning phase: a table where the classification was spread over to states where there are no prior examples, and that allows the classification of other images.
To show the applicability of this porposal, experiments and results obtained with this method are presented in the next section.

Figure 7.3: Objects segmented from test images of the dataset.

7.4 Experiments and Results

Several experiments were executed using the ASL dataset. Each experiment consists of two processing phases: the training of the RL and the execution phase, where the training quality can be verified. To train the RL, we used 36 test images, from which approximately 160 images containing objects were segmented (using the *floodcanny* algorithm in Chapter 6) and previously classified. Samples of the segmented images are presented in Figure 7.3. Furthermore, 360 background images, also resulting from the segmentation process, were used. The training was performed according to the algorithm described in the previous section.

To evaluate the result of the learning process, the Leave-One-Out method was used. Using this method, the RL selection module was trained with all the test images but one, and then the image left out (that can contain several regions of interest (ROIs) with objects and background) is used to test the result of the learning. This test phase corresponds to the execution phase, which can be used on the real robot during on-line exploration of the environment, and its working diagram is presented in Figure 7.4.

Six different experiments were conducted, using three different combinations of image attributes as state space descriptors and two different image sizes (the image original size and a 10 by 10 pixels reduced size image). The combinations of image attributes used as state space are: mean and standard deviation of the image intensity (MS); mean and standard deviation of the image intensity plus entropy of the image (MSE); and mean and standard deviation of the image intensity plus the number of interest points detected by the Difference of Gaussians operator (DoG).

The rewards used during the learning phase were computed using a set of training images. Figure 7.5 shows part of the reward table built for the first stage of the

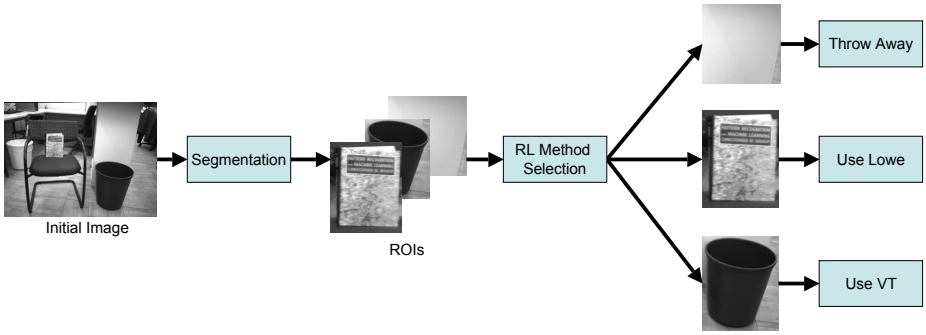


Figure 7.4: Execution Phase of the system.

first experiment (MS). It is a 100 x 130 figure, where the rows represent the possible values of mean intensity of the image, and the columns represents 100 possible values in standard deviation of the image intensity (the MS space). There are three kinds of states, marked in the image by: "X", which indicates that in the training set exists an image with this combination of mean and std dev values, "·", which corresponds to points that represent images that do not contain objects (backgrounds) and the rest of the space, which is left without any marking and corresponds to points in the MS state where there is no information about it. As it can be seen, this is a sparse image.

During the learning phase (described in Section 7.3), if the learning agent reaches a state labeled as "X", it receives the value +10 and if it reaches a state labeled as "·", it receives a reward of -10. Otherwise the reward is zero.

Figure 7.6 shows the results of applying the RL algorithm during the learning phase. As it can be seen, the classification was spread over to states where there are no prior examples, allowing the classification of other images. This table is the one used during the execution phase.

Tables 7.1 and 7.2 present the results obtained for the six experiments. The first row of Table 7.1 shows the percentage of times that the agent correctly chose to discard a background image, and the second row shows the percentage of times the agent correctly chose to use the Lowe algorithm, instead of the Vocabulary Tree one. The columns in this table present the results for the six experiments, the first three using the original image and, from the fourth to sixth column, showing the results for the reduced size image. The last column shows the percentage of times a human expert takes the correct action. Table 7.2 is similar to Table 7.1, but shows the classification error. The first row shows the percentage of images discarded as background, when they should be analyzed, and the second row presents the number of times the Lowe algorithm is chosen, when the correct one is the Vocabulary Tree.

The results show that the use of Reinforcement Learning to decide which algorithm should be applied to recognize objects yields good results, for all different combinations of image attributes used. Furthermore, in some cases, Reinforce-

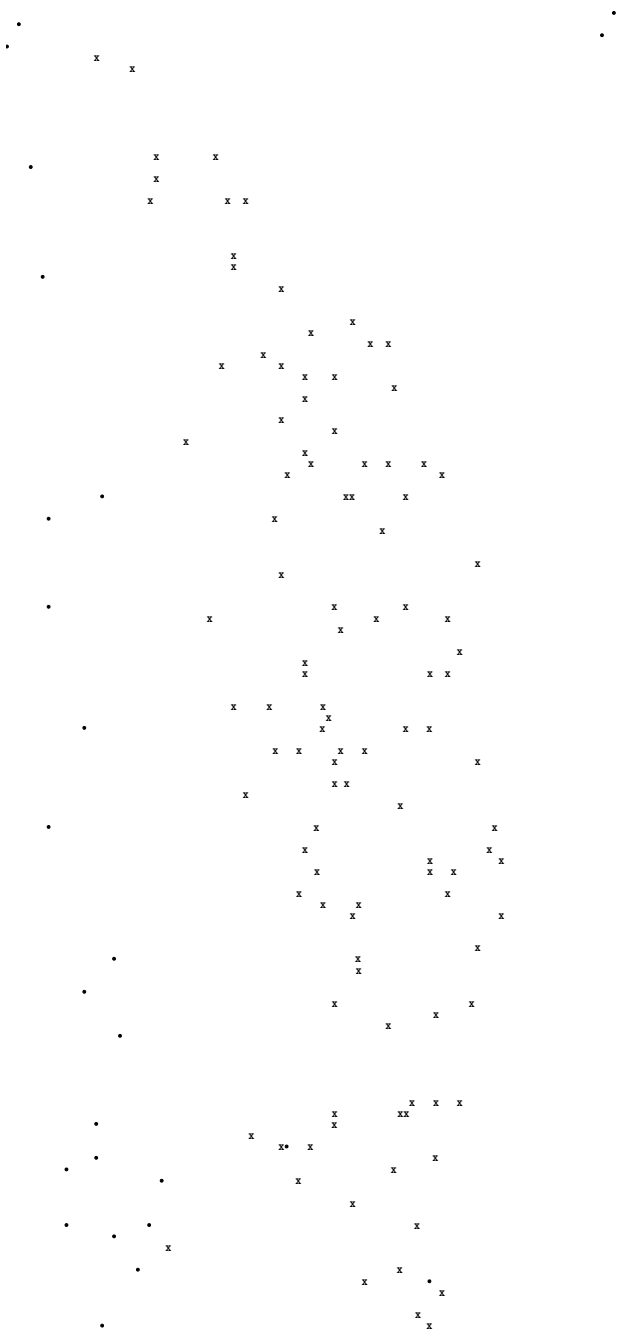


Figure 7.5: Image of the reward table built for the first experiment (Mean and Standard deviation of image intensity space).

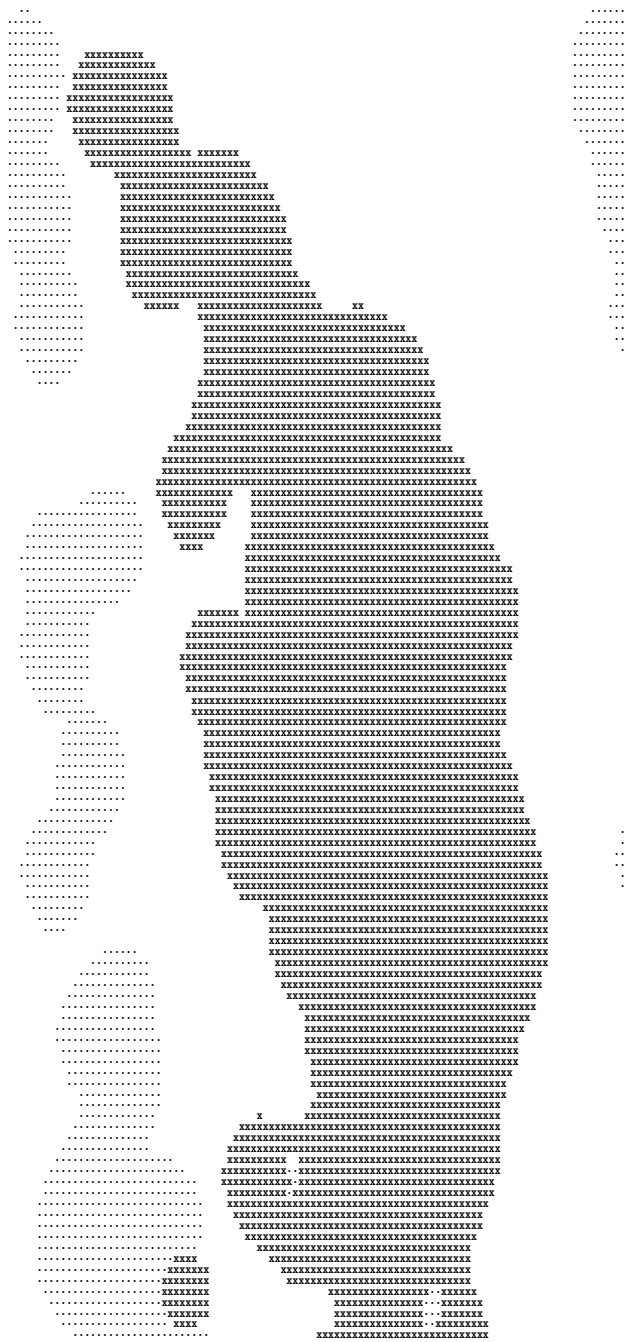


Figure 7.6: Classification table learned in the first experiment.

Table 7.1: Correctly classified images (percentage)

	Full Img			Small Img			Expert
	MS	MSE	DoG	MS	MSE	DoG	
Back	91.9	100.0	98.0	92.6	100.0	98.9	100.0
Lowe	84.5	100.0	44.4	76.0	98.4	38.1	93.2

Table 7.2: Incorrect classification (percentage)

	Full Img			Small Img			Expert
	MS	MSE	DoG	MS	MSE	DoG	
Back	12.8	1.8	14.2	20.4	2.4	25.3	8.2
Lowe	11.6	1.9	7.9	15.8	1.9	9.9	10.8

ment Learning performed better than a human expert.
These tables also show that the best combination of attributes was mean and standard deviation of the image intensity plus entropy of the image (MSE), which presented very good results for original size images as well as reduced size ones. On the other hand, the use of the number of interest points detected by the Difference of Gaussians operator as state space did not produce good results, failing to choose Lowe's algorithm more than half of the time.
Reinforcement Learning algorithms were implemented in C and experiments were executed on a Pentium 4 Computer running Ubuntu Linux and a PowerMac G4 running Mac OS X. The Reinforcement Learning parameters used in the experiments were: the learning rate $\alpha = 0.1$ and the discount factor $\gamma = 0.9$. Values in the Q table were randomly initiated.

Chapter 8

Conclusion and Future Work

In this thesis we have addressed the issues of robot global localization and object recognition. Our contributions are described in chapters 3 to 7 and are summarized in this chapter.

In Chapter 3 we have proposed and evaluated a signature to characterize places that can be used for global localization. This signature consists of a constellation of feature descriptors, computed from affine-covariant regions, extracted from a panoramic image that has been acquired in the place we want to add to the map. Later, these signatures are compared to the constellation extracted from a new panoramic image using geometric constraints, and the most similar signature is selected as the current location. To compare the different signatures, the 4-point algorithm with RANSAC to reject false matches is used. Combinations of feature detectors have been shown to perform best if combined with adequate mechanisms, such as reciprocal matching or distance to the second nearest neighbor, to reject incorrect pairings of features before computing the essential matrix.

Regarding the validation of the global localization schema, the results obtained show that by using the combination of different feature detectors, a room can be reliably recognized in indoor environments from a distance of up to 4 meters from the point where the reference panorama was obtained. The best results (90% correct localizations) were achieved by combining all the three evaluated detectors.

Moreover, we have also compared the results of our proposed affine-covariant region detectors approach with the scale-invariant region detectors methodology proposed in Lowe (2004), widely used in robot navigation, and we have shown that the affine-covariant regions outperformed Lowe's scale-invariant method.

In order to speed-up the otherwise very expensive descriptor matching phase, a global similarity technique usually employed for object recognition, the Vocabulary Tree from Nister and Stewenius (2006), has been effectively applied

to re-rank the map nodes for a given query panorama and save most of the computation time.
Furthermore, we tested how the performance degrades if only a conventional perspective image is used instead of an omnidirectional image. Results of a 10 repetitions experiment with random 45° sections (with a minimum amount of texture) from all the test panoramas show a surprisingly good performance.

To complement the proposed global localization system, in Chapter 4 we investigated the applicability of a homing method to travel from one map node to another. Also, when the robot has several likely hypotheses about its current location (i.e. room) homing can be used to return to the position where the most likely panorama from the database was made. If the hypothesis with the highest probability was correct, then the panorama at that location should be more similar to the panorama from the database, making it the significantly best hypothesis. In the case where an incorrect panorama was chosen from the database, then the same steps should be taken as before in order to find out in which room the robot is.
Although there are several methods to do homing, such as the 1D method of Hong et al (1991), warping (Franz et al, 1998) or snapshots (Lambrinos et al, 2000), the *ALV homing* method (Lambrinos et al, 1998, 2000) has been used mainly because of its simplicity and low computational complexity.
In order to evaluate the proposed method, initial experiments using a simulated environment were conducted and later it was tested in a real world scenario. The real world experiments were done with panoramas acquired in three different rooms at the IIIA research center.
The locations at which the robot acquired the panoramas were measured manually and used to calculate the ground truth homing directions, which were then used to verify the homing method results. Features were extracted from panoramic images to be used by the homing method. Two invariant feature detectors were tested: Difference of Gaussians extrema (DoG) by Lowe (2004) and the Maximally Stable Extremal Regions (MSER) by Matas et al (2002). Only the horizontal location of the features was used, i.e. the cylindrical angle, and not height, nor scale, because it is not necessary for the method.
The ALV homing was found to be a good working method, however it performed worse in rooms where the width and length differ greatly. This has been explained by the way the features are projected on the panorama and by the *equal distance assumption* (Franz et al, 1998).
Vardy (2005) discusses biologically based homing methods in his thesis and also did experiments with several of them. In his work he used panoramas that were made with a camera pointed to a parabolic mirror. The advantage of acquiring a panorama like this is the speed of acquisition, whereas with our method first images from several angles had to be retrieved and then stitched to create a high resolution panorama. In order to compare both methods of panorama acquisition additional experiments using Vardy's data sets were performed using the SIFT and MSER features.

When comparing the results of IIIA data sets and Vardy's data sets we can see that the ALV homing method performs slightly better on the IIIA data sets, but the difference is not significant. There might be several reasons to explain this, such as the difference in resolution, the camera or the environment. For these reasons it cannot be concluded that having a panorama with a higher resolution made by rotating a camera or a camera ring is much better for our proposed ALV homing approach than using a camera and a parabolic mirror. On the other hand, the faster acquisition of the parabolic panoramas might be more important than the slightly better performance.

Regarding the feature types, in our experiments MSER significantly outperformed SIFT. Also Mikolajczyk et al (2005) confirmed that MSER is one of the most robust feature detectors. The artificial landmarks in the *robotics laboratory* were used to compare the local feature approach with the more traditional artificial landmarks. The results with the artificial landmarks were significantly better than with the invariant features, the error was about 7° less than using the MSER detector (in only the lower half of the panorama). However, this difference seems low enough to justify the applicability of the presented homing method, because does not require setting up the environment by placing artificial landmarks.

Although equipping a robot with robust methods to extract semantic information from perceptual data is of utmost importance in order to have truly cognitive robots capable of realizing complex tasks in complex environments, we are aware of few works addressing this. Consequently, Chapters 5 and 6 of this thesis address this gap by comparing and improving two state of the art object recognition methods with a focus on making them useful for mobile robotics.

In order to test the object recognition method we have created a challenging dataset of video sequences with our mobile robot while moving in an office type environment. These sequences have been acquired at a resolution of 640×480 pixels with the robot cameras, and are full of blurred images due to motion, large viewpoint and scale changes and object occlusions.

In Chapter 5 we have evaluated the SIFT object recognition method, proposed by Lowe (2004). Issues such as training image quality, approximate local descriptor matching or false hypotheses filtering methods are evaluated in a subset of the proposed dataset. Furthermore, we propose and evaluate several modifications to the original schema to increase the detected objects and reduce the computational time. The parameter settings that attained best overall results are subsequently tested in the rest of the dataset and carefully evaluated to have a clear picture of the response that can be expected from the method with respect to untextured objects or image degradations. Next, a similar evaluation is carried on for the second method, the Vocabulary Tree proposed by Nister and Stewenius (2006).

From the results obtained, it can be seen that with the present implementation of the methods, the SIFT object recognition method adapts better to the performance requirements of a robotics application. Furthermore, it is easy to train,

since a single good quality image sufficed to attain good recall and precision levels. However, although this method is resistant to occlusion and reasonable levels of motion blur, its usage is mostly restricted to flat well textured objects. Also, classification (generalizing to unseen object instances of the same class) is not possible with this approach.
On the other hand, the Vocabulary Tree method has obtained good recognition rates both for textured and untextured objects, but too many false positives per image were found.

Since the two evaluated object recognition methods showed complementary properties, in Chapter 7 we have proposed a Reinforcement Learning approach to select the most appropriate method for a new image or sub-window. Based on simple attributes extracted on-line from the images, such as mean intensity and intensity deviation, the method is able to decide which object recognition system has the best probability of reliably detecting an object in the new image, or if the image has no interesting information and should be discarded.
The results obtained show that the use Reinforcement Learning to decide which algorithm should be used to recognize objects yields good results, performing better than a human expert in some cases. To the best of our knowledge, there is no similar approach using automatic selection of algorithms for object recognition.

Future Work

There are many lines in which the work done in this thesis can be continued, especially the second part deserves much more research effort.
The global localization method proposed in Chapter 3 must be tested with larger scale databases to further assess its robustness. Also detectors that concentrate on other types of features should be evaluated, like the Maximally Stable Color Regions by Forssén (2007) or the Stable Symmetry Features by Huebner and Zhang (2006). Regarding the feature-based approach to homing presented in Chapter 4, the next step is to evaluate it in real navigation tasks. For this, the robot will have to be equipped with a compass, because an external orientation reference to align the panoramas is required by the method. The next step would be testing the whole system in indoor navigation experiments.
For the object recognition experiments, only the SIFT and the Shape Context descriptors have been evaluated, future work should include testing other image descriptors, especially color-based ones. Chromatic information is usually disregarded in computer vision research, but it is obvious that including it in the object representation would help improve the recognition results.
Although we have evaluated the proposed object recognition methods in a wide range of dimensions, one that is lacking is a more in-depth study of how the composition and size of the training set affects the overall results. For example, having similar objects, as the different monitors or chairs in the IIIA30 dataset,

can cause confusion to the methods. Therefore future work should address the evaluation of different sub-sets of target objects.

The main limitation of the SIFT object recognition method is that only the first nearest neighbor of each test image feature is considered in the subsequent stages. This restriction makes the SIFT method very fast, but at the same time makes it unable to detect objects with repetitive textures. Other approaches with direct matching, like that of Leibe et al (2008), overcome this by allowing every feature to vote for all feasible object hypotheses given the feature position and orientation. Evaluating this type of methods, or modifying the SIFT to accept several hypotheses for each test image feature, would be an interesting line of continuation.

The heuristics proposed in this work to improve the SIFT object recognition method have been manually designed. Nevertheless, it would be much better if the system itself was able to learn and generalize which bounding boxes parameters constitute valid hypotheses, for instance using Reinforcement Learning. This constitutes a research line on its own that we are very interested in following.

The sliding windows approach could be improved by allowing windows with a good probability of a correct detection to inhibit neighboring and/or overlapping windows, or simply keeping the best window for a given object would clearly reduce the number of false positives.

Regarding the segmentation schema, we believe that results can be improved by adopting more reliable techniques, able to resists highlights and shadows. Besides, textured areas pose a problem to the segmentation algorithm as, with the current technique, no windows will be casted in scattered areas. using a Monte Carlo approach to fuse neighboring regions may help alleviate the problem without significantly affecting the computational time. Also a voting mechanism to detect areas with a high number of small regions can be attempted.

In each candidate region detected by the segmentation method, a set of windows is evaluated. However, we used only square windows, and the results of the different windows of the set were not combined or used in a voting process to decide the most probable object hypothesis.

Furthermore, combining the intensity and disparity segmentations could help improve the accuracy of the detected regions. Also it would allow fusing the multiple small regions found in areas with high information content, where the intensity segmentation method fails to find a large enough region. Future work towards evaluating these alternatives should therefore be undertaken.

Despite the drawbacks found with the Vocabulary Tree approach, *bag of features* techniques are a current topic of research, and improved approaches are constantly being presented. An example is the one proposed by Lampert et al (2008), that seems able to overcome the problems encountered in this work. Therefore, evaluating this method with the IIIA30 sequence is a natural continuation line of this thesis.

Using the Reinforcement Learning method developed in Chapter 7 to discard areas of the image that are likely to contain no objects is also a clear continuation

line of this thesis. The evaluation of sub-windows with this technique is much faster than the Vocabulary Tree, and could therefore lead to a speed-up of the overall method and a reduction of false positives.

Bibliography

Agarwal A, Triggs B (2008) Multilevel image coding with hyperfeatures. International Journal of Computer Vision 78(1):15–27

Angeli A, Filliat D, Doncieux S, Meyer JA (2008) A fast and incremental method for loop-closure detection using bags of visual words. IEEE Transactions On Robotics, Special Issue on Visual SLAM pp 1031–1036

Baumberg A (2000) Reliable feature matching across widely separated views. In: Conference on Computer Vision and Pattern Recognition (CVPR 2000), Hilton Head, SC, USA, pp 1774–1781

Bay H, Ess A, Tuytelaars T, Gool LV (2008) Surf: Speeded up robust features. Computer Vision and Image Understanding (CVIU) 110(3):346–359

Bhanu B, Peng J (2000) Adaptive integrated image segmentation and object recognition. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 30(4):427–441, DOI 10.1109/5326.897070

Booij O, Zivkovic Z, Krose B (2006) From sensors to rooms. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop - From Sensors to Human Spatial Concepts, Beijing, China, pp 53–58

Booij O, Terwijn B, Zivkovic Z, Krose B (2007) Navigation using an appearance based topological map. In: IEEE International Conference on Robotics and Automation (ICRA), pp 3927–3932

Borotschnig H, Paletta L, Prantl M, Pinz A (1999) Appearance-based active object recognition. Image and Vision Computing 18(9):715–728

Brown M, Lowe D (2003) Recognising panoramas. In: ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision, IEEE Computer Society, Washington, DC, USA, p 1218

Brown M, Lowe D (2005) Unsupervised 3D object recognition and reconstruction in unordered datasets. 3-D Digital Imaging and Modeling, 2005 3DIM 2005 Fifth International Conference on pp 56–63

Burgard W, Fox D, Jans H, Matenar C, Thrun S (1999) Sonar-based mapping of large-scale mobile robot environments using em. In: ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 67–76

Busquets D (2003) A multiagent approach to qualitative navigation in robotics. PhD dissertation, Universitat Politècnica de Catalunya

Busquets D, Sierra C, de Mantaras RL (2003) A multiagent approach to qualitative landmark-based navigation. Autonomous Robots 15(2):129–154

Cabani C, MacLean W (2006) A Proposed Pipelined-Architecture for FPGA-Based Affine-Invariant Feature Detectors. Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop pp 121–121

Canny J (1986) A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6):679–698

Carwright BA, Collet TS (1983) Landmark learning in bees: Experiments and models. Journal of Comparative Physiology 151:521–543

Choset H, Nagatani K (2001) Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. IEEE Transactions On Robotics and Automation 17(2):125–137

Csurka G, Bray C, Dance C, Fan L (2004) Visual categorization with bags of keypoints. Workshop on Statistical Learning in Computer Vision, ECCV pp 1–22

Cummins M, Newman P (2007) Probabilistic appearance based navigation and loop closing. In: Proc. IEEE International Conference on Robotics and Automation (ICRA07), pp 2042–2048

Cummins M, Newman P (2008) Accelerated appearance-only slam. In: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, pp 1828–1833

Darrell T (1998) Reinforcement learning of active recognition behaviors," interval research technical report 1997-045. (http://www.interval.com/papers/1997-045 - portions of this paper previously appeared. In: in Advances in Neural Information Processing Systems 8, (NIPS '95, MIT Press, pp 858–864

Darrell T, Pentland A (1996a) Active gesture recognition using learned visual attention. In: Touretzky DS, Mozer MC, Hasselmo ME (eds) Advances in Neural Information Processing Systems, The MIT Press, vol 8, pp 858–864

Darrell T, Pentland A (1996b) Active gesture recognition using partially observable markov decision processes. Pattern Recognition, 1996, Proceedings of the 13th International Conference on 3:984–988 vol.3, DOI 10.1109/ICPR.1996.547315

Davison A, Reid I, Molton N, Stasse O (2007) Monoslam: Real-time single camera slam. IEEE Transactions on Pattern Analysis and Machine Intelligence pp 1052–1067

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B 39:1–38

Doucet A, de Freitas N, Murphy KP, Russell SJ (2000) Rao-blackwellised particle filtering for dynamic bayesian networks. In: UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 176–183

Draper BA, Bins J, Baek K (1999) ADORE: Adaptive object recognition. In: International Conference on Vision Systems, pp 522–537

Duda RO, Hart PE, Stork DG (2001) Pattern Classification, 2nd edn. John Wiley

Edelman S, Intrator N, Poggio T (1997) Complex cells and object recognition, unpublished manuscript, University of Cornell

Ekvall S, Jensfelt P, Kragic D (2006) Integrating Active Mobile Robot Object Recognition and SLAM in Natural Environments. Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on pp 5792–5797

Elfes A (1989) Occupancy grids: a probabilistic framework for robot perception and navigation. PhD thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University

Elfes A (1990) Sonar-based real-world mapping and navigation. Autonomous robot vehicles pp 233–249

Engelson S (1994) Passive map learning and visual place recognition. PhD thesis, Yale University

Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, vol 2, pp 264–271

Ferrari V, Fevrier L, Jurie F, Schmid C (2008) Groups of adjacent contour segments for object detection. IEEE Trans Pattern Anal Mach Intell 30(1):36–51

Filliat D, Meyer J (2003) Map-based navigation in mobile robots - I. a review of localisation strategies. Journal of Cognitive Systems Research 4(4):243–282

Forssén PE (2007) Maximally stable colour regions for recognition and matching. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, IEEE, Minneapolis, USA

Franz MO, Schölkopf B, Mallot HA, Bülthoff HH (1998) Where did i take that snapshot? scene-based homing by image matching. Biological Cybernetics 79:191–202

Freksa C, Moratz R, Barkowsky T (2000) Schematic maps for robot navigation. In: Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications, Springer-Verlag, London, UK, pp 100–114

Fulkerson B, Vedaldi A, Soatto S (2008) Localizing objects with smart dictionaries. In: Forsyth DA, Torr PHS, Zisserman A (eds) ECCV (1), Springer, Lecture Notes in Computer Science, vol 5302, pp 179–192

Gächter S, Harati A, Siegwart R (2008) Incremental object part detection toward object classification in a sequence of noisy range images. In: ICRA, IEEE, pp 4037–4042

Galindo C, Saffiotti A, Coradeschi S, Buschka P, Fernandez-Madrigal J, Gonzalez J (2005) Multi-hierarchical semantic maps for mobile robotics. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.(IROS 2005), pp 2278–2283

Goldhoorn A (2008) Solving ambiguity in global localization of autonomous robots. Master's thesis, University of Groningen

Goldhoorn A, Ramisa A, de Mántaras RL, Toledo R (2007a) Robot homing simulations using the average landmark vector method. Tech. Rep. RR-IIIA-2007-03, IIIA-CSIC, Bellaterra

Goldhoorn A, Ramisa A, de Mntaras RL, Toledo R (2007b) Using the average landmark vector method for robot homing. In: 19th International Conference of the ACIA, IOS Press, Frontiers in Artificial Intelligence and Applications, vol 163, pp 331–338

Gordon I, Lowe D (2006) What and Where: 3D Object Recognition with Accurate Pose. Lecture Notes in Computer Science 4170:67

Grauman K, Darrell T (2005) The pyramid match kernel: Discriminative classification with sets of image features. In: ICCV, IEEE Computer Society, pp 1458–1465

Hafner V, Moller R (2001) Learning of Visual Navigation Strategies. In: European Workshop on Learning Robots (EWLR), Prague, pp 47–56

Hafner VV (2001) Adaptive homing: Robotic exploration tours. Adaptive Behavior 9(3/4):131–141

Harris C, Stephens M (1988) A combined corner and edge detector. In: 4th Alvey Vision Conference, pp 147–151

Hartley R, Zisserman A (2004) Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, ISBN: 0521540518

Heymann S, Maller K, Smolic A, Froehlich B, Wiegand T (2007) SIFT implementation and optimization for general-purpose GPU. Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision

Hong J, Tan X, Pinette B, Weiss R, Riseman E (1991) Image-based homing. In: Proceedings of the 1991 IEEE International Conference on Robotics and Automation, pp 620–625

Hossain I, Liu J, You J (1999) Tropical cyclone pattern recognition for intensity and forecasting analysis from satellite imagery. Systems, Man, and Cybernetics, 1999 IEEE SMC '99 Conference Proceedings 1999 IEEE International Conference on 6:851–856 vol.6, DOI 10.1109/ICSMC.1999.816663

Huebner K, Zhang J (2006) Stable Symmetry Feature Detection and Classification in Panoramic Robot Vision Systems. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 3429–3434

Jegou H, Harzallah H, Schmid C (2007) A contextual dissimilarity measure for accurate and efficient image search. In: Procedings of the Computer Vision and Pattern Recognition conference, pp 1–8

Kadir T, Zisserman A, Brady M (2004) An affine invariant salient region detector. In: Computer Vision - ECCV 2004, 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 3021, pp 228–241

Kaufman L, Rousseeuw P (1990) Finding groups in data: an introduction to cluster analysis. John Wiley & Sons

Ke Y, Sukthankar R (2004) PCA-SIFT: a more distinctive representation for local image descriptors. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, vol 2, pp II–506–II–513 Vol.2

Kortenkamp D (1993) Cognitive maps for mobile robots: A representation for mapping and navigation. PhD thesis, University of Michigan

Kuipers B (1978) Modeling spatial knowledge. In: Cognitive Science: A Multidisciplinary Journal, vol 2, pp 129–153

Lambrinos D, Möller R, Pfeifer R, Wehner R (1998) Landmark navigation without snapshots: the average landmark vector model. In: Elsner N, Wehner R (eds) Proc 26th Göttingen Neurobiology Conference, Thieme-Verlag, p 221

Lambrinos D, Möller R, Labhart T, Pfeifer R, Wehner R (2000) A mobile robot employing insect strategies for navigation. Robotics and Autonomous Systems 30(1-2):39–64

Lampert CH, Blaschko MB, Hofmann T (2008) Beyond sliding windows: Object localization by efficient subwindow search. In: CVPR, IEEE Computer Society

Leibe B, Leonardis A, Schiele B (2004) Combined object categorization and segmentation with an implicit shape model. Workshop on Statistical Learning in Computer Vision, ECCV pp 17–32

Leibe B, Leonardis A, Schiele B (2008) Robust object detection with interleaved categorization and segmentation. International Journal of Computer Vision 77(1-3):259–289

Lepetit V, Pilet J, Fua P (2004) Point matching as a classification problem for fast and robust object pose estimation. In: CVPR (2), pp 244–250

Levenberg K (1944) A method for the solution of certain non-linear problems in least squares. Q Appl Math 2(2):164–168

Lindeberg T (1998) Feature detection with automatic scale selection. International Journal of Computer Vision 30(2):79–116

Lindeberg T, Gårding J (1997) Shape-adapted smoothing in estimation of 3-D shape cues from affine deformations of local 2-D brightness structure. Image Vision Comput 15(6):415–434

Lowe D (1999) Object recognition from local scale-invariant features. In: ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2, IEEE Computer Society, Washington, DC, USA, p 1150

Lowe D (2001) Local feature view clustering for 3d object recognition. In: CVPR (1), IEEE Computer Society, pp 682–688

Lowe D (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2):91–110

Lu F, Milios E (1997) Globally consistent range scan alignment for environment mapping. Auton Robots 4(4):333–349, DOI http://dx.doi.org/10.1023/A:1008854305733

Marszalek M, Schmid C (2007) Accurate object localization with shape masks. In: CVPR, IEEE Computer Society

Matas J, Chum O, Martin U, Pajdla T (2002) Robust wide baseline stereo from maximally stable extremal regions. In: Rosin PL, Marshall D (eds) Proceedings of the British Machine Vision Conference, BMVA, London, UK, vol 1, pp 384–393

Matas J, Chum O, Urban M, Pajdla T (2004) Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing 22(10):761–767

Mikolajczyk K (2002) Detection of local features invariant to affines transformations. PhD thesis, INPG, Grenoble

Mikolajczyk K, Schmid C (2002) An affine invariant interest point detector. Proc ECCV 1:128–142

Mikolajczyk K, Schmid C (2004) Scale & affine invariant interest point detectors. Int J Comput Vision 60(1):63–86

Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis & Machine Intelligence 27(10):1615–1630

Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, Gool L (2005) A comparison of affine region detectors. International Journal of Computer Vision 65:43–72(30)

Minut S, Mahadevan S (2001) A reinforcement learning model of selective visual attention. In: AGENTS '01: Proceedings of the fifth international conference on Autonomous agents, ACM, New York, NY, USA, pp 457–464

Mitchell T (1997) Machine Learning. McGraw Hill, New York

Möller R (2000) Insect visual homing strategies in a robot with analog processing. Biological Cybernetics, special issue: Navigation in Biological and Artificial Systems 83:231–243

Möller R, Lambrinos D, Roggendorf T, Pfeifer R, Wehner R (2001) Insect strategies of visual homing in mobile robots. In: Webb B, Consi TR (eds) Biorobotics. Methods and Applications, AAAI Press / MIT Press, pp 37–66

Moosmann F, Nowak E, Jurie F (2008) Randomized clustering forests for image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(9):1632–1646

Moravec H (1980) Obstacle avoidance and navigation in the real world by a seeing robot rover. In: tech. report CMU-RI-TR-80-03, Robotics Institute, Carnegie Mellon University & doctoral dissertation, Stanford University, Stanford University Stanford, CA, USA, available as Stanford AIM-340, CS-80-813 and republished as a Carnegie Mellon University Robotics Institue Technical Report to increase availability

Moravec H (1988) Sensor fusion in certainty grids for mobile robots. AI Magazine 9(2):61–74

Mori G, Belongie S, Malik J (2005) Efficient shape matching using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(11):1832–1837

Muja M, Lowe D (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Applications (VISAPP'09)

Murillo A, Sagüés C, Guerrero J, Goedemé T, Tuytelaars T, Van Gool L (2007) From omnidirectional images to hierarchical localization. Robotics and Autonomous Systems 55(5):372–382

Murphy-Chutorian E, Aboutalib S, Triesch J (2005) Analysis of a biologically-inspired system for real-time object recognition. Cognitive Science Online 3(2):1–14

Mutch J, Lowe D (2006) Multiclass object recognition with sparse, localized features. In: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol 1, pp 11–18

Nieto J, Guivant J, Nebot E (2004) The hybrid metric maps (hymms): A novel map representation for DenseSLAM. In: Proceedings of the 2003 IEEE International Conference on Robotics and Automation, ICRA 2004, IEEE Computer Society, New Orleans, USA, pp 391–396

Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. Conf Computer Vision and Pattern Recognition 2:2161–2168

Nowak E, Jurie F, Triggs B (2006) Sampling Strategies for Bag-of-Features Image Classification. European Conference on Computer Vision, ECCV p 490

O'Keefe J, Dostrovsky J (1971) The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat. In: Brain research, Elsevier/North-Holland Biomedical Press, vol 34, pp 171–175

Opelt A, Pinz A, Fussenegger M, Auer P (2006a) Generic object recognition with boosting. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(3):416–431

Opelt A, Pinz A, Zisserman A (2006b) A boundary-fragment-model for object detection. Proc ECCV 2:575–588

Opelt A, Pinz A, Zisserman A (2006c) Incremental learning of object detectors using a visual shape alphabet. Proc CVPR 1(3-10):4

Paletta L, Pinz A (2000) Active object recognition by view integration and reinforcement learning. Robotics and Autonomous Systems 31(1–2):71–86

Paletta L, Prantl M, Pinz A (1998) Reinforcement learning of 3-d object recognition from appearance. In: CONALD'98: Proceedings of the 1998 Conference on Automated Learning and Discovery

Paletta L, Fritz G, Seifert C (2005) Q-learning of sequential attention for visual object recognition from informative local descriptors. In: ICML '05: Proceedings of the 22nd International Conference on Machine Learning, ACM, New York, NY, USA, pp 649–656

Peng J, Bhanu B (1998a) Closed-loop object recognition using reinforcement learning. Pattern Analysis and Machine Intelligence, IEEE Transactions on 20(2):139–154, DOI 10.1109/34.659932

Peng J, Bhanu B (1998b) Delayed reinforcement learning for adaptive image segmentation and feature extraction. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 28(3):482–488, DOI 10.1109/5326.704593

Philbin J, Chum O, Isard M, Sivic J, Zisserman A (2007) Object retrieval with large vocabularies and fast spatial matching. In: Proc. Conference on Computer Vision and Pattern Recognition, pp 1–8

Pinto N, Cox DD, Dicarlo JJ (2008) Why is real-world visual object recognition hard? PLoS Computational Biology 4(1):151–156

Rabinovich A, Vedaldi A, Galleguillos C, Wiewiora E, Belongie S (2007) Objects in Context. Computer Vision, 2007 ICCV 2007 IEEE 11th International Conference on pp 1–8

Ramisa A (2006) Qualitative navigation using panoramas. Msc thesis, Universitat Autònoma de Barcelona

Ramisa A, Tapus A, de Mantaras RL, Toledo R (2008a) Mobile Robot Localization using Panoramic Vision and Combination of Local Feature Region Detectors. Proc IEEE International Conference on Robotics and Automation, Pasadena, California pp 538–543

Ramisa A, Vasudevan S, Scaramuzza D, de Mántaras RL, Siegwart R (2008b) A tale of two object recognition methods for mobile robots. In: Gasteratos A, Vincze M, Tsotsos JK (eds) ICVS, Springer, Lecture Notes in Computer Science, vol 5008, pp 353–362

Rummery GA, Niranjan M (1994) On-line Q-learning using connectionist systems. Tech. Rep. CUED/F-INFENG/TR 166, Cambridge University Engineering Department

Sahba F, Tizhoosh HR, Salama MMMA (2008) A reinforcement agent for object segmentation in ultrasound images. Expert Syst Appl 35(3):772–780, DOI http://dx.doi.org/10.1016/j.eswa.2007.07.057

Scaramuzza D, Siegwart R (2008) Appearance guided monocular omnidirectional visual odometry for outdoor ground vehicles. IEEE Transactions on Robotics, Special Issue on Visual SLAM. In press. Guest editors: Jose' Neira, Andrew Davison, John J. Leonard, Publication date: October 2008

Se S, Lowe D, Little J (2001) Vision-based mobile robot localization and mapping using scale-invariant features. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, pp 2051–2058

Serre T, of Brain D, Sciences C, of Technology MI (2006) Learning a dictionary of shape-components in visual cortex: Comparison with neurons, humans and machines. Massachusetts Institute of Technology, Dept. of Brain and Cognitive Sciences

Serre T, Wolf L, Bileschi SM, Riesenhuber M, Poggio T (2007) Robust object recognition with cortex-like mechanisms. IEEE Trans Pattern Anal Mach Intell 29(3):411–426

Shi J, Tomasi C (1994) Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on, pp 593–600

Shokri M, Tizhoosh HR (2003) Using reinforcement learning for image thresholding. Electrical and Computer Engineering, 2003 IEEE CCECE 2003 Canadian Conference on 2:1231–1234 vol.2, DOI 10.1109/CCECE.2003.1226121

Shokri M, Tizhoosh HR (2004) Q(lambda)-based image thresholding. In: CRV '04: Proceedings of the 1st Canadian Conference on Computer and Robot Vision, IEEE Computer Society, Los Alamitos, CA, USA, pp 504–508

Shokri M, Tizhoosh HR (2008) A reinforcement agent for threshold fusion. Appl Soft Comput 8(1):174–181, DOI http://dx.doi.org/10.1016/j.asoc.2006.12.003

Shum H, Szeliski R (1997) Panoramic image mosaics. Tech. Rep. MSR-TR-97-23, Microsoft Research

Sivic J, Zisserman A (2003) Video Google: a text retrieval approach to object matching in videos. Computer Vision, 2003 Proceedings Ninth IEEE International Conference on pp 1470–1477

Smith R, Self M, Cheeseman P (1990) Estimating uncertain spatial relationships in robotics. Autonomous robot vehicles pp 167–193

Smith RC, Cheeseman P (1986) On the representation and estimation of spatial uncertainty. International Journal of Robotics Research 5(4):56–68

Sutton R, Barto AG (1998) Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA

Szeliski R, Shum HY (1997) Creating full view panoramic image mosaics and environment maps. In: SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, vol 31, pp 251–258

Tapus A, Siegwart R (2006) A cognitive modeling of space using fingerprints of places for mobile robot navigation. In: Proceedings the IEEE International Conference on Robotics and Automation (ICRA), Orlando, U.S.A., May 2006, pp 1188–1193

Tapus A, Heinzer S, Siegwart R (2004) Bayesian programming for topological global localization with fingerprints. In: IEEE International Conference on Robotics and Automation (ICRA), New Orleans, USA, vol 1, pp 598–603

Taylor GW (2004) Reinforcement Learning for Parameter Control of Image-Based Applications. MSc Thesis, University of Waterloo, Ontario, Canada

Thrun S (1998) Learning metric-topological maps for indoor mobile robot navigation. Artificial Intelligence 99(1):21–71

Thrun S (2002) Robotic mapping: A survey. In: Lakemeyer G, Nebel B (eds) Exploring Artificial Intelligence in the New Millenium, Morgan Kaufmann

Thrun S, Burgard W, Fox D (1998) A probabilistic approach to concurrent mapping and localization for mobile robots. Machine Learning 5(3-4):253–271

Thrun S, Beetz M, Bennewitz M, Burgard W, Creemers A, Dellaert F, Fox D, Hahnel D, Rosenberg C, Roy N, Schulte J, Schulz D (2000) Probabilistic algorithms and the interactive museum tour-guide robot minerva. International Journal of Robotics Research 19(11):972–999

Tizhoosh H, Taylor G (2006) Reinforced contrast adaptation. International Journal of Image and Graphics 6(3):377–392

Tolman EC (1948) Cognitive maps in rats and men. Psychological Review 55(4):198–208

Tomatis N, Nourbakhsh I, Siegwart R (2002) Hybrid simultaneous localization and map building: Closing the loop with multi-hypotheses tracking. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation, ICRA, IEEE Computer Society, Washington, DC, USA, pp 2749–2754

Torralba A, Murphy K, Freeman W, Rubin M (2003) Context-based vision system for place and object recognition. Computer Vision, 2003 Proceedings Ninth IEEE International Conference on pp 273–280

Usher K, Ridley P, Corke P (2003) Visual servoing of a car-like vehicle-an application of omnidirectional vision. In: Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on, vol 3, pp 4288–4293

Uyttendaele M, Eden A, Szeliski R (2001) Eliminating ghosting and exposure artifacts in image mosaics. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, vol 2, pp 509–516

Valgren C, Lilienthal AJ (2008) Incremental spectral clustering and seasons: Appearance-based localization in outdoor environments. In: Proc. IEEE Int. Conf. on Robotics and Automation, pp 1856–1861

Vardy A (2005) Biologically plausible methods for robot visual homing. PhD dissertation, Carleton University

Vasudevan S (2008) Spatial cognition for mobile robots: A hierarchical probabilistic concept-oriented representation of space. PhD thesis, Autonomous Systems Lab (ETHZ)

Vasudevan S, Gächter S, Nguyen V, Siegwart R (2007) Cognitive maps for mobile robots - an object based approach. Robotics and Autonomous Systems 55(5):359–371

Vazquez E, van de Weijer J, Baldrich R (2008) Image Segmentation in the Presence of Shadows and Highlights. In: Computer Vision-Eccv 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV, Springer, pp 1–14

Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. In: Proceedings of the Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, vol 1, pp 511–518

Watkins CJCH (1989) Learning from Delayed Rewards. PhD Thesis, University of Cambridge

Wehner R (1994) The polarization-vision project: championing organismic biology. Progr Zool (Fortschr Zool) 39:103–143

Wersing H, Kirstein S, Schneiders B, Bauer-Wersing U, Körner E (2008) Online learning for bootstrapping of object recognition and localization in a biologically motivated architecture. In: Gasteratos A, Vincze M, Tsotsos JK (eds) ICVS, Springer, Lecture Notes in Computer Science, vol 5008, pp 383–392

Whitehead SD, Ballard DH (1991) Learning to perceive and act by trial and error. Mach Learn 7(1):45–83, DOI http://dx.doi.org/10.1023/A:1022619109594

Wolf C, Jolion JM (2003) Extraction and recognition of artificial text in multimedia documents. Pattern Anal Appl 6(4):309–326, DOI http://dx.doi.org/10.1007/s10044-003-0197-7

Yin PY (2002) Maximum entropy-based optimal threshold selection using deterministic reinforcement learning with controlled randomization. Signal Process 82(7):993–1006, DOI http://dx.doi.org/10.1016/S0165-1684(02)00203-7

Yu X, Yi L, Fermuller C, Doermann D (2007) Object Detection Using A Shape Codebook. Unpublished

Zeki S (2001) Localization and globalization in conscious vision. Annual Review Neuroscience 24:57–86

Zhang J, Marszalek M, Lazebnik S, Schmid C (2007) Local features and kernels for classification of texture and object categories: A comprehensive study. International Journal of Computer Vision 73(2):213–238

Monografies de l'Institut d'Investigació en Intel·ligència Artificial

Num. 1 J. Puyol, *MILORD II: A Language for Knowledge–Based Systems*

Num. 2 J. Levy, *The Calculus of Refinements, a Formal Specification Model Based on Inclusions*

Num. 3 Ll. Vila, *On Temporal Representation and Reasoning in Knowledge–Based Systems*

Num. 4 M. Domingo, *An Expert System Architecture for Identification in Biology*

Num. 5 E. Armengol, *A Framework for Integrating Learning and Problem Solving*

Num. 6 J. Ll. Arcos, *The Noos Representation Language*

Num. 7 J. Larrosa, *Algorithms and Heuristics for Total and Partial Constraint Satisfaction*

Num. 8 P. Noriega, *Agent Mediated Auctions: The Fishmarket Metaphor*

Num. 9 F. Manyà, *Proof Procedures for Multiple-Valued Propositional Logics*

Num. 10 W. M. Schorlemmer, *On Specifying and Reasoning with Special Relations*

Num. 11 M. López-Sánchez, *Approaches to Map Generation by means of Collaborative Autonomous Robots*

Num. 12 D. Robertson, *Pragmatics in the Synthesis of Logic Programs*

Num. 13 P. Faratin, *Automated Service Negotiation between Autonomous Computational Agents*

Num. 14 J. A. Rodríguez, *On the Design and Construction of Agent-mediated Electronis Institutions*

Num. 15 T. Alsinet, *Logic Programming with Fuzzy Unification and Imprecise Constants: Possibilistic Semantics and Automated Deduction*

Num. 16 A. Zapico, *On Axiomatic Foundations for Qualitative Decision Theory - A Possibilistic Approach*

Num. 17 A. Valls, *ClusDM: A multiple criteria decision method for heterogeneous data sets*

Num. 18 D. Busquets, *A Multiagent Approach to Qualitative Navigation in Robotics*

Num. 19 M. Esteva, *Electronic Institutions: from specification to development*

Num. 20 J. Sabater, *Trust and Reputation for Agent Societies*

Num. 21 J. Cerquides, *Improving Algorithms for Learning Bayesian Network Classifiers*

Num. 22 M. Villaret, *On Some Variants of Second-Order Unification*

Num. 23 M. Gómez, *Open, Reusable and Configurable Multi-Agent Systems: A Knowledge Modelling Approach*

Num. 24 S. Ramchurn, *Multi-Agent Negotiation Using Trust and Persuasion*

Num. 25 S. Ontañon, *Ensemble Case-Based Learning for Multi-Agent Systems*

Num. 26 M. Sánchez, *Contributions to Search and Inference Algorithms for CSP and Weighted CSP*

Num. 27 C. Noguera, *Algebraic Study of Axiomatic Extensions of Triangular Norm Based Fuzzy Logics*

Num. 28 E. Marchioni, *Functional Definability Issues in Logics Based on Triangular Norms*

Num. 29 M. Grachten, *Expressivity-Aware Tempo Transformations of Music Performances Using Case Based Reasoning*

Num. 30 I. Brito, *Distributed Constraint Satisfaction*

Num. 31 E. Altamirano, *On Non-clausal Horn-like Satisfiability Problems*

Num. 32 A. Giovannucci, *Computationally Manageable Combinatorial Auctions for Supply Chain Automation*

Num. 33 R. Ros, *Action Selection in Cooperative Robot Soccer using Case-Based Reasoning*

Num. 34 A. García-Cerdaña, *On some Implication-free Fragments of Substructural and Fuzzy Logics*

Num. 35 A. García-Camino, *Normative Regulation of Open Multi-agent Systems*

Num. 36 A. Ramisa Ayats, *Localization and Object Recognition for Mobile Robots*

Num. 37 C.G. Baccigalupo, *Poolcasting: an intelligent technique to customise music programmes for their audience*

Num. 38 J. Planes, *Design and Implementation of Exact MAX-SAT Solvers*

Num. 39 A. Bogdanovych, *Virtual Institutions*

Num. 40 J. Nin, *Contributions to Record Linkage for Disclosure Risk Assessment*

Num. 41 J. Argelich Romà, *Max-SAT Formalisms with Hard and Soft Constraints*

Num. 42 A. Casali, *On Intentional and Social Agents with Graded Attitudes*

Num. 43 A. Perreau de Pinnick Bas, *Decentralised Enforcement in Multiagent Networks*