UNIVERSITY OF SOUTHAMPTON

# Biological and Metric Maps Applied to Robot Homing

by

Sotirios Ch. Diamantas

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

June 2010

UNIVERSITY OF SOUTHAMPTON

## ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Sotirios Ch. Diamantas

Navigation is at the heart of today's mobile robotic systems. The use of landmarks for the guidance of autonomous systems is an essential part of the process. Shape, size, and colour are some of the properties of the landmarks. In this work biological and other methods have been developed for navigating a simulated mobile robot back to its home. Optical flow which is not a property of the landmarks but a property of the camera motion, has been used for building topological maps which are used to localise a robot. The advantage of this approach is that there is no need for storing or retrieving images. Any computation is performed on the basis of vectors. Metric maps are generated using a parsimonious localisation and mapping algorithm using a laser range finder.

The purpose of this research is to effectively navigate a robot to its home position using computationally efficient biological and other techniques for navigation. Biology is seen as an alternative solution to the problems robots' encounter which include algorithmic complexity, performance, and power consumption among others. Biological inspiration provides simple, yet effective methods for the solutions of such problems. The careful examination of such methods has twofold gain. The study of the principles of biological organisms entails making better autonomous systems that will, in turn help us perceive and understand better the underlying mechanisms of biological organisms.

It is therefore essential to have an understanding of how biological and robotic systems work. This work discusses the techniques and strategies found both in animals and robots. It then goes on to describe the implementation of a series of simulation algorithms inspired by biology and other fields that assist in the navigation process, and in particular, homing. Their results are discussed and analysed. The novelty of this research lies in its methods for robot homing that make use of optical flow information to recognise a location as well as methods based on a laser range finder to avoid obstacles, perform path planning, localise a robot, and map the environment.

# Contents

# List of Figures

# List of Tables

# Nomenclature

**Chapter 1**

| | |
|---|---|
| $SLAM$ | Simultaneous Localisation and Mapping |
| $TBL$ | Turn-Back-and-Look |

**Chapter 2**

| | |
|---|---|
| $ALV$ | Average Landmark Vector |
| $r$ | uniqueness of landmark |
| $g'$ | local minimum of valley method |
| $g$ | global minimum of valley method |
| $\delta$ | threshold of valley method |
| $r_l$ | reliability of landmark $l$ |
| $l$ | landmark |
| $n$ | number of steps in TBL |
| $H$ | home location |
| $v_h$ | home vector |
| $v_t^i$ | tangential vector |
| $v_r^i$ | radial vector |
| $ALV_C$ | current snapshot |
| $ALV_S$ | snapshot at goal location |
| $EKF$ | Extended Kalman Filter |
| $RFID$ | Radio Frequency Identification |
| $DP$ | Dynamic Programming |
| $PLRF$ | Panorama Laser Range Finder |
| $k$ | time instant |
| $i$ | landmark |
| $\mathbf{x}_k$ | state vector at time $k$ |
| $\mathbf{u}_k$ | control vector at time $k$ |
| $\mathbf{m}_i$ | vector for location of $i$th landmark |
| $\mathbf{z}_{ik}$ | robot observation for the location of $i$th landmark at time $k$ |
| $\mathbf{X}_{0:k}$ | history of robot locations |

| | |
|---|---|
| $\mathbf{U}_{0:k}$ | history of control inputs |
| $\mathbf{m}$ | set of all landmarks |
| $\mathbf{Z}_{0:k}$ | set of all landmark observations |
| $\mathbf{G}_k$ | matrix for mapping control inputs into a predicted state space |
| $\mathbf{P}_k$ | observation update in Kalman Filter |
| $\mathbf{v}_k$ | innovation |
| $\mathbf{S}_k$ | covariance of innovation |
| $\mathbf{h}$ | a transformation of representations used by sensors into state estimates |
| $\nabla\mathbf{h}$ | Jacobian of h |
| $\mathbf{W}_k$ | Kalman weight matrix |
| $\mathbf{R}_k$ | covariance |
| $\mathbf{f}(\cdot)$ | vehicle kinematics |
| $\mathbf{Q}_k$ | covariance matrix of the process noise |
| $UAV$ | Unmanned Aerial Vehicle |
| $LK$ | Lucas-Kanade algorithm |
| $H$ | Hessian image |
| $I$ | intensity of pixel |
| $\partial x$ | horizontal displacement |
| $\partial y$ | vertical displacement |
| $M$ | autocorrelation matrix |
| $u$ | x component of u (pixel) |
| $v$ | y component of v (pixel) |

**Chapter 3**

| | |
|---|---|
| $ODE$ | Open Dynamics Engine |

**Chapter 4**

| | |
|---|---|
| $VFH$ | Vector Field Histogram |
| $r$ | radius of robot |
| $C_h$ | x component of circle $C$ |
| $C_k$ | y component of circle $C$ |
| $R_x$ | x component of robot position $R$ |
| $R_y$ | y component of robot position $R$ |
| $\Delta$ | Discrminant |
| $R_C$ | central ray of laser finder |
| $R_{C_n}$ | selected path |
| $A$ | set of maximum range values of laser finder |

| | |
|---|---|
| $ray_x$ | ray x |
| $d$ | Euclidean distance |
| $L$ | landmark |
| $L_x^i$ | x component of landmark location |
| $L_y^i$ | y component of landmark location |
| $R_{S_L}$ | left side ray of laser scanner |
| $R_{S_R}$ | right side ray of laser ray |
| $\theta$ | angle between robot and landmark |
| $\Omega$ | set of snapshots |
| $r^n$ | robot position at which a snapshot is taken |
| $v_n$ | a point on a line perpendicular to $\theta$ |
| $t$ | time interval |
| $\epsilon$ | noise in $\theta$ |
| $C$ | 2 × 2 matrix |
| $h$ | estimated position of landmark |
| $n$ | number of observations |
| $f_D$ | distance function between robot and landmark |
| $d$ | a value of distance |
| $f_E$ | velocity function of robot |
| $e$ | a value of velocity |
| $E(D)$ | expected distance |
| $a$ | constant |
| $b$ | coefficient |

**Chapter 5**

| | |
|---|---|
| $SAD$ | Sum of Absolute Differences |
| $NCC$ | Normalised Cross Correlation |
| $T_{x,y,z}$ | translation between robot and landmark |
| $R_{r,p,y}$ | rotation angles (roll, pitch, yaw) |
| $h-FOV$ | horizontal Field of View |
| $d$ | Euclidean distance between pixels |
| $R_f$ | reference pixel |
| $C_j$ | pixel on circumference |
| $r$ | radius between reference pixel and circumference pixel |
| $I$ | image |
| $T$ | template |
| $n$ | number of observations |
| $f_D$ | distance function between robot and landmark |
| $d$ | a value of distance |

| | |
|---|---|
| $f_E$ | velocity function of robot |
| $e$ | a value of velocity |
| $k$ | vector |
| $x_k, y_k$ | centre point of vector k |
| $\bar{x}, \bar{y},$ | mean position of all vectors (mean of the means) |
| $\chi_k$ | Euclidean distance between $\bar{x}, \bar{y}$ and $x_k, y_k$ |
| $\delta$ | Euclidean distance between the mean of the means of two snapshots |
| $f_X$ | log-normal probability density function (pdf) |
| $F_X$ | log-normal cumulative density function (cdf) |
| $erfc$ | complementary error function |
| $\Phi$ | standard normal cdf |
| $P_\delta$ | probability of log-normal distribution between two set of vectors |
| $min_i$ | landmark $i$ with minimum number of vectors |
| $max_j$ | landmark $j$ with with maximum number of vectors |
| $P_T$ | similarity score |
| $t$ | time frame of a snapshot |

# Acknowledgements

I would like to express my gratitude to my supervisors Dr Richard Crowder and Dr Klaus-Peter Zauner for all their help, guidance, and understanding all these difficult years for me. I am also grateful to plenty of IAM and university people. In particular, I am grateful to Ioannis Vetsikas, Tasos Oikonomidis, Georgios Chalkiadakis, Johnsen Kho, and Renato Cavalcante. I would like to thank all my friends for all their support.

Last, but not least I am deeply grateful to my parents and my brother for everything they have sacrificed in their life for me. Unfortunately, I will never be able to give them back what they have lost for me.

*To my parents Charalampos and Eleni, and to my brother Christos.*

*where you dedicate your dreams, there you belong to.*

# Chapter 1

# Introduction

Recently, there has been a resurgence of interest in the use of biologically-inspired robotics due to the simple, yet effective strategies they employ. A honeybee can estimate its velocity and its distance to an object by calculating the rate of change of objects seen by its retina. Optical flow techniques that calculate the rate of change of objects in a camera have been developed and used for collision avoidance and robot centring when passing through a corridor. These types of systems have adopted techniques and methods that are found in biological organisms with simple sensing and neural complexity. Nevertheless, the behaviour they exhibit is effective and, in the majority of cases quite complex. Hence, a considerable amount of research interest has been undertaken in mobile robotic systems inspired by nature; a rather new area with a large number of applications from surveillance and inspection systems to systems that can be used in medicine, in space missions, or military applications.

From the rich information provided by the visual sensors, this research is aiming at the high-level vision part, in particular extracting meaningful features, such as landmarks.

> *A landmark is a salient feature of the world. They can be divided into global and local, and natural or artificial. Global, refer to landmarks such as the sun, the moon, the stars while local refer to a river, a tree, or a rock. An artificial landmark can be a building, a bridge, or a corner of a wall.*

Based on studies on biological organisms, particularly insects, the use of landmarks play an important role in their search for food, a process called foraging, as well as for returning to their home (nest or hive for ants or bees, respectively), a method called homing. Homing is an important part of the navigational process. A robot may need to return to its base for a number of reasons like recharging of batteries, failure of a subsystem, or completion of a task. This work is also looking at different navigation strategies and how these can be combined to aid a mobile robot in navigation. Detection of large-scale landmarks during the beginning of the outbound trip as well as the use of optical flow for building maps are some of the strategies developed for homing.

## 1.1 Motivation and Objectives

The research question of this work is to find how a mobile robot can effectively return to its home using computationally efficient biological and other techniques for navigation. The hypothesis is that insects, and in particular honeybees use optic flow for localisation and map building. The initial work consists of a literature review in the areas of visual navigation of animals and, in particular, insects, followed by a review of the Simultaneous Localisation and Mapping (SLAM) literature. A number of simulations have been conducted to support the homing methods and techniques, the results from which are presented in this thesis.

Motivation for this work has raised from the fact that robotics and computer vision will be playing more and more significant role in the years to come. They are large disciplines and their application areas are numerous ranging from surveillance systems to systems used in difficult or dangerous for the human environments. Moreover, by studying the principles of biological organisms efficient and useful robotic systems can be built that will, in turn, help us to perceive and understand better the underlying mechanisms that govern navigation in animals.

The aim of this work is to apply vision and laser systems to mobile robotics for the purpose of providing efficient navigation capabilities to robotic systems. Such systems can operate on various types of environments, both indoor and outdoor. The application areas of robots capable of performing homing, that is the return to the home or base, are wide ranging. Homing is of importance for robots that are used for search and rescue. Areas that have been hit by earthquakes make up such environments. Yet, regions that suffer from extreme meteorological phenomena form another candidate for this type of systems. Planetary missions can also provide an excellent application area. Matsuno and Tadokoro (2004) provide an overview of this type of agents that are currently developed in Japan for search and rescue in hazardous and disastrous environments caused by earthquakes and other natural phenomena.

Having set the aim of this research, the following objectives need to be met:

- Add to the understanding of visual-based navigation for fully autonomous mobile robots. Inspiration is drawn from biological and engineering studies, with the overriding objective of application with the search and rescue field. To claim this, the research is looking at the effectiveness of novel algorithms that will aid the robot navigation.

- Methods based on visual landmark recognition are computationally expensive. Another objective of this research is to provide robust yet computationally efficient methods for robot navigation. For this reason, inspiration from biology has provided the methods for landmark recognition and topological map building. Inspiration from biology has a two-fold gain as it can provide insights for robust robotic systems while at the same time it can explain the underlying mechanisms that underpin biological organisms.

- Model and simulate biological navigational methods that are applicable to robotic systems. For this objective, this research will be looking at navigational methods of well-studied animals, that is, insects such as ants and honeybees. Apart from the fact that these insects have provided a plethora of information to the research community, their navigational methods are efficient and effective for the small number of neurons they possess.

- Traditional methods for localisation and mapping are rather accurate but their computational cost is quite high. This is a core problem encountered in almost all methods for localising and mapping a robot within an environment. In the SLAM methodology computational cost increases quadratically with the number of landmarks. For this reason, localisation and mapping using visual methods inspired by biology are used in conjunction with methods based on laser scanner for efficient path localisation and map building.

The aim of this work is not to make a complete model of a biological organism, but rather to take inspiration from biological that will meet the objectives set. To complement the work, a study in homing has been conducted from the engineering point of view. For the purpose of this work, the main sensing capability used is vision and a laser range scanner is used for path planning, and localisation and mapping. The reason for this is that vision systems are relatively low cost, provide a plethora of information, and are used across the robotics community. However, the greatest disadvantage of vision systems lies in the processing of the information. Though an image can provide an enormous amount of information, the time needed for the processing is critical, especially for real-time systems. In addition, image noise is a major drawback of vision systems this impedes and obstructs the proper processing. This leads, sometimes, to erroneous information being extracted from the image.

Furthermore, a laser sensor is employed for obstacle avoidance and path planning. A laser range finder is highly suitable for that purpose due to its strength in estimating distances accurately while unburdening the system from problems arising in other sensors like infrared or sonar. In fact, there is a large number of works that utilise laser range sensors for obstacle avoidance (Ulrich and Borenstein, 1998; An and Wang, 2004; Koren and Borenstein, 1991; Martinez et al., 1998).

This thesis makes significant contributions to the state of the art that can be categorised into two parts. The first part that is inspired by biology and, the second part that uses methods found in engineering.

- Optical flow has been used to build topological map based on the optical flow 'fingerprint' of the landmarks. The main novelty of this approach is that it does not make use of any information apart from the optical flow patterns of the landmarks. Velocity, position, and distance between the robot and the landmark are all remain unknown. Optical flow, however, is not a property of the landmarks, like colour, shape, and size, but a property of the camera motion. Based on a training algorithm landmarks can be recognised based on their optical flow 'signature'. In addition, during the initiation of the outbound trip the robot is performing the TBL method in order to detect large-scale landmarks which are located close to its home position. These landmarks will be used for recognition of the home position when the robot is completing its inbound trip.
- The second part of this work involves building metric maps based on a laser range finder. The algorithm detects corners in the environment, that act as landmarks, which are used for localising the robot with respect its environment. A camera has also been used to support this method in cases where there are no distinct landmarks. Yet, a comparison is made between the optical flow method and a least squares approach in estimating depth, that is the distance between the robot and a landmark. A mixed approach has also been implemented where least squares is combined with optical flow in inferring depth. However, there is a trade off between robustness and computational cost.

Figure 1.1 shows a general plan that has been followed for the accomplishments of these aims and objectives. The research questions have been defined based on the readings of literature review. This, in turn, has provided an input to the study of the visual navigation strategies of animals, mainly insects, and of the robots currently used. A large number of robots, nowadays, make use of the SLAM (Simultaneous Localisation and Mapping) for navigational purposes. Some others try to model biological organisms and mimic their abilities; thus, the emergence of biorobotics field. Inspiration from the biological and the engineering world has provided input for the methods to be followed and the algorithms developed for this work. Experiments have been conducted using primarily a 3D simulator, Gazebo (2006), and a 2D simulator, Player/Stage (2006) which are part of the Gerkey et al. (2003) project. MATLAB (2005), has also been used for data analysis. Therefore, using simulation tools the robot is capable of building topological and metric maps. During the outbound trip the robot performs a series of strategies that have been inspired by the way animals navigate. These include the turn-back-and-look (TBL) approach that detects large scale landmarks while retreating from home position and the optical flow based landmark recognition. On the other hand a method which relates to SLAM methodology has been employed for metric map building and localisation and mapping using a laser range finder. The outcome of all those is the navigation of a mobile robot, in particular homing, within an unknown environment.

A number of publications have been produced from this research. The following list shows the papers and the conferences at which have and will be appeared.

- S. Ch. Diamantas, A. Oikonomidis, and R. M. Crowder. Towards optical flow-based robotic homing. In Proceedings of the International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Barcelona, Spain, July 2010 (In Press)b.
- S. Ch. Diamantas, A. Oikonomidis, and R. M. Crowder. Depth estimation for autonomous robot navigation: A comparative approach. In Proceedings of the International Conference on Imaging Systems and Techniques, Thessaloniki, Greece, July 2010 (In Press)a.
- S. Ch. Diamantas and R. M. Crowder. Localisation and mapping using a laser range finder: A goal-seeking approach. In Proceedings of the Fifth International Conference on Autonomic and Autonomous Systems, pages 270-276, Valencia, Spain, April 2009.

## 1.2 Structure of the Thesis

The thesis comprises of six chapters. Chapter 1 provides an introduction to the field and a theoretical background to the areas directly related to the subject of the research, namely navigation of insects and machines. It also discusses the application areas of robots performing homing and provides the general plan followed for the attainment of the aims and objectives.

Chapter 2 provides an introduction to navigation techniques, employing vision as the main modality, used by animals and robots. It also presents related work of other researchers in biology as well as in SLAM, and the various methodologies they have used. Moreover, it presents the basic models developed and tested on robotic systems which have been inspired by insects and by engineering studies. It also presents the mathematical formulation of the optic flow algorithm.

Chapter 3 describes the general simulation platform. The architecture of the simulator is discussed as well as the various tools that encompass the simulation environment. A description of the related two simulators (2D and 3D) is given.

Chapter 4 deals with the evaluation of the preparations that have been made. The methods and their results are presented and analysed. It also gives quantitative information

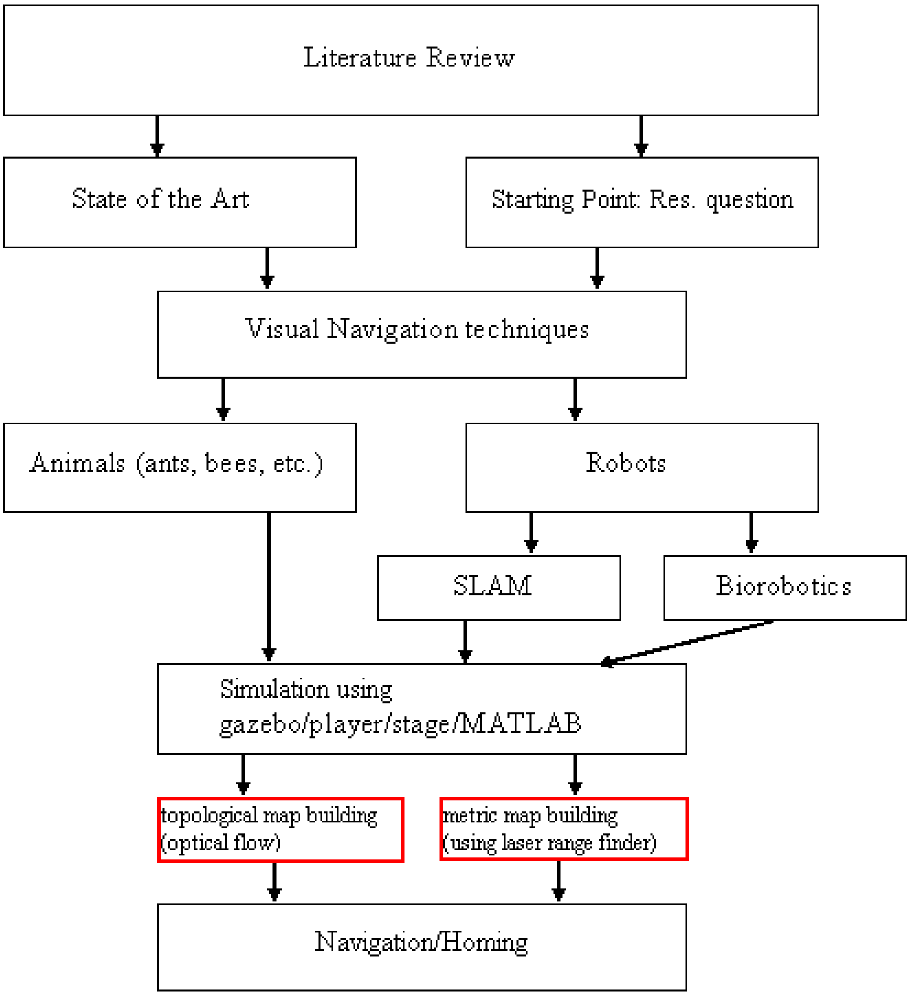

Literature Review
State of the Art
Starting Point: Res. question
Visual Navigation techniques
Animals (ants, bees, etc.)
Robots
SLAM
Biorobotics
Simulation using gazebo/player/stage/MATLAB
topological map building (optical flow)
metric map building (using laser range finder)
Navigation/Homing

FIGURE 1.1: Plan of the research programme. The initial phase has included literature review that has pointed out the research question of the work. It has followed by studies in insect and robot navigation. The simulation tools have been used for the implementation of the optical flow-based map building and the modelling and simulation of a laser range finder for metric map building and localisation. The red boxes depict the contribution of this research.

regarding the recognition of landmarks by the robot using the optic flow algorithm, and the large-scale landmark recognition and detection algorithms. Finally, it shows how effectively a robot can localise with respect to a landmark based on the topological maps it has built.

Chapter 5 presents the engineering studies of localisation and mapping as well as the algorithm developed for creating metric maps of the environment. The algorithm of this chapter uses a laser range scanner which is then supported by a camera to produce better results.

Chapter 6 summarises the work achieved thus far and provides a section for further research. It discusses the important areas where research can continue and provides some of their limitations.

# Chapter 2

# Background Work

## 2.1 Introduction

This chapter considers the visual navigation strategies employed by animals. Most of the discussion will be restricted to insects, such as ants and bees that have been well-studied. However, a brief discussion is performed around the visual methods other animals use, such as birds. In addition, this chapter surveys the various models of visual homing that have been developed based on biological studies. Furthermore, various strategies that ants employ and which are related to the current work of visual navigation in unknown environments. These strategies are path integration, visual landmarks, and searching. Searching is a strategy that is employed when all other methods have failed. The literature review of this chapter acts as the basis for the developed models which are explained in the following two chapters. In this chapter the two most popular biological models of homing are presented, namely ALV and snapshot as well as the utilisation of optical flow algorithms on robotic systems. The limitations of each of the available algorithms are highlighted, thus indicating the novel work that could be developed. Moreover, the chapter surveys the navigation methods used in today's robotic systems from the engineering point view, namely Simultaneous Localisation and Mapping (SLAM) and its mathematical underpinning. Finally it concludes, with an explanation of the Lucas-Kanade optic flow algorithm that has been used for building topological maps based on the optical flow 'fingerprint' of the landmarks.

## 2.2 Historical Perspective

In the decade of 1940's, Norbert Wiener coined the term cybernetics that refers to the communication and control in the animal and the machine (Wiener, 1965), this was the first link between intelligence in animals and machines. Biologically inspired robotics

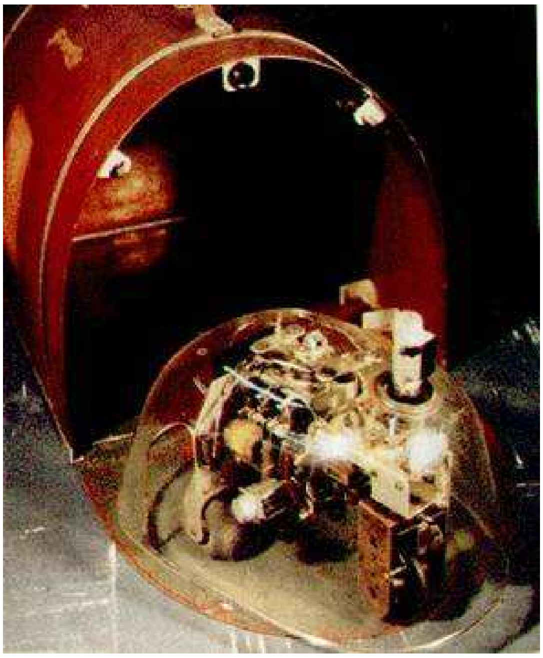


Figure 2.1: Grey Walter's tortoise constructed between late 40's and early 50's. Figure from The Grey Walter Online Archive (2006).

is a rather new approach but its roots can be traced as early as the beginning of the twentieth century when Hammond built a heliotrope based on Loeb's theory of tropism* (Sharkey, 2002). In this theory, Loeb suggested that animals be attracted and repelled by stimuli similar to that found in plants.

The first practical mobile robots used principles such as phototaxis (light) and were built by mid of the twentieth century by the neurophysiologist Grey Walter (Holland, 2003). Grey Walter used only two receptors, a light and a touch sensor, for his tortoises (Fig. 2.1). These simple machines were able to exhibit complex behaviour when interacting with the environment. Few of the behaviours that his machines exhibited were social organisation, free-will, pertinacity, recognition of self, and others (Holland, 2003). As lately as the 80's, Braitenberg (1984) described various robots that exhibit complex behaviour based on simple rules. These robots could control the speed of the motor based on a sensor; they could be attracted or repelled by light, or perform various trajectories between light sources. All of these approaches had a common element, they exhibited life-like behaviours based on a few simple neurons. Inspired by these simple interactions Brooks (1985, 1991) invented the behaviour-based approach on robotics. Behaviour-based robotics came as an alternative to the traditional approaches of artificial intelligence where a robot was required to have a precise representation of its environment, a detailed map for example.

*Tropism and taxis refer to the same thing, i.e., the attraction or repulsion of an agent directed by a stimulus. Some examples include chemotaxis, geotaxis, phototaxis, phonotaxis, and osmotropotaxis - trail following.

The work presented in this thesis is to take inspiration from the biology world, for these simple reasons stated above. The simplicity and ease of processing that complex tasks require make insects, and animals in general, perfect candidates for modelling and simulation.

## 2.3 Biological Navigation and Homing

This section provides an overview of the capabilities and methods animals employ for navigation. Insects are the animals whose navigational behaviour is examined in this thesis, since they are well-studied and exhibit a variety of interesting capabilities as compared to their physical and neural structure. In particular, this section's discussion is about ants and honeybees. However, other animals' navigational behaviour is briefly reported. Concerning navigation, it is well known that certain animals can cover very long distances, returning back home and traversing the same route with a high degree of accuracy. An insect's brain contains approximately $10^6$ neurons, a small number compared to the complex tasks, such as navigation, they can carry out and also when compared to vertebrates (Collett and Collett, 2002).

Insects have developed efficient methods that do not require complicated calculations for solving those tasks from which inspiration has been taken. Such an example is when robots calculate their position relative to some landmarks by performing complex calculations such as feature extraction, sensor fusion, etc. On the other hand as discussed briefly in Chapter one, studies have shown that honeybees use optic flow, that is, the rate of change of image motion in their retina that makes them locate at any instant of time their position relative to some goal location or landmarks Srinivasan et al. (1999). This, however, does not seem to be the case for the desert ant *Cataglyphis fortis* (Ronacher et al., 2000) where they seem to rely mainly on idiothetic means. The discovery of the use of image motion in the retina of honeybees has led to the use of optic flow techniques in robotic systems.

Two other techniques that animals employ for navigation and which are of interest for this work are path integration and landmark recognition. These two techniques along with searching are discussed below. Yet, another popular technique employed by insects is pheromone trail following. It is used by many species of ants, but not from all. The well-studied desert ant *Cataglyphis* does not lay pheromone since this species lives in deserts where the high temperatures would cause an evaporation of trailings (Wehner et al., 1996). Nevertheless, even the ant species that lay pheromone have been shown (Klotz, 1987; Harrison et al., 1989) to associate trails with visual landmarks. Thus, making homing more robust and accurate. When there exist both pheromone trails and visual landmarks, experienced ants adopt the landmark strategy whereas the inexperienced ants are directed by pheromone trails (Harrison et al., 1989).

Healy (1998) discusses a large number of similar, as well as different navigation strategies employed by fish, pigeons, mammals, and birds. Hafner (2004) demonstrates how the hippocampus area of rats assists in cognitive mapping of the environment. She introduces various navigational strategies that are employed by animals, and specifically, rats. In particular, the question posed is to what extent navigation behaviour in animals is innate or learnt. Experiments (Hafner, 2004) have shown that in principle visual homing strategies can be both learnt and evolved by artificial agents. Even a sparse topological representation of place cells can lead to good spatial representation of the environment where metric information can easily be extracted, if required, by the agent. Nevertheless, it is not clear which navigation strategy is applied by an animal, since its behaviour consists of a combination of different strategies. When, and to what extent, the different strategies are chosen and which sensory modalities are applied is still an open question.

## 2.4 Insect Strategies

This section discusses the major strategies that animals employ when foraging or homing. A significant work by Lehrer and Bianco (2000) examines the turn-back-and-look (TBL) behaviour that honeybees exhibit (Lehrer, 1991, 1993; Fry and Wehner, 2005). Figure 2.2 depicts the TBL behaviour that bees perform. In this experiment, when bees come out of a reward feeding site they turn by 180° and inspect the feeding site again and again while they are retreating. When bees were captured on their departure from a feeding site, they were not able to find the goal location, while those who were not, performed the TBL behaviour and found the food source on the next arrival (Lehrer, 1991, 1993). The experiments showed also that colour and shape of landmarks are mainly learnt during arrival to a feeding site. However, when bees are prevented from learning these cues on arrival, they learn them on departure from the feeding site. Thus, colour and shape are not learnt during the TBL phase.

It seems that bees during the TBL approach learn the spatial relationship of landmarks and of the feeding site. This can be explained by the fact that nearer objects move faster than further ones. This hypothesis was tested by Lehrer and Collett (1994) using a large and a small cylinder as landmarks. The bees trained during the arrival preferred the cylinder with the memorised size. The same occurred with the TBL approach. Thus, showing that bees learn the size of a landmark during the turn-back-and-look approach. In another experiment, bees could view either the large or the small cylinder on both arrivals and departures. When bees tests at the end of the TBL phase showed a preference to the correct distance over the correct size. When the same bees were tested with no TBL phase, they preferred the cylinder with the correct size. This reveals that information about absolute distance occurs only at the initial phase to a feeding site,

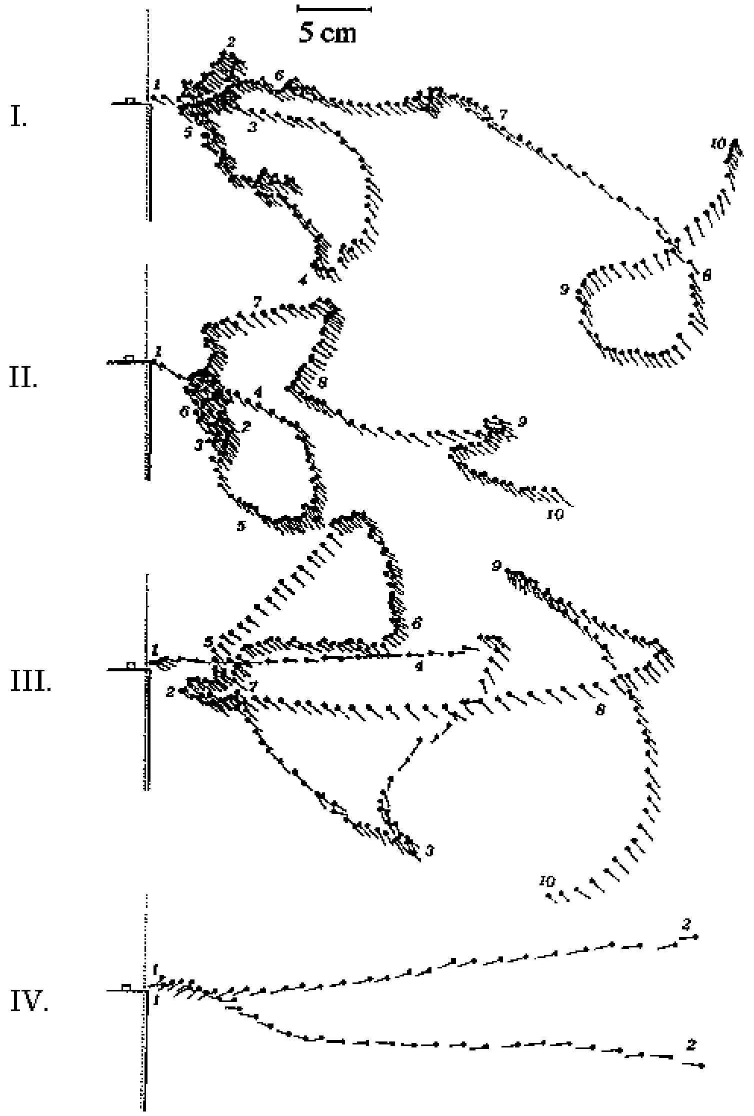


Figure 2.2: The turn-back-and-look behaviour of insects. Numbers along the trajectories depict the course of flight of a honeybee while departing from its hive. **I-III.** The first TBL of three individual bees. **IV.** Departure of two individual bees after the TBL-phase has ended. Figure from Lehrer and Bianco (2000).

since at this phase bees are keen on finding near landmarks to the feeding site. Once this has occurred, the bees try to learn the angular size of the landmark.

A similar work to the TBL approach with ants, was performed by Judd and Collett (1998). In their work, Judd and Collett (1998), showed that wood ants take several snapshots of a familiar beacon from different vantage points. Their hypothesis is based on the fact that the retinal image of an object changes according to the viewing distance and sometimes direction. In fact, the rate of change of a retinal image changes more rapidly when the viewing distance is getting smaller and smaller between the object and the viewer. On the other hand, there is not much rate of change of the object stored on the retina when the viewing distance is becoming larger and larger. In other words, the rate of change of retinal image is dependent on the distance as well as on the velocity. An object's rate of change image may change rapidly because distance between the insect and object is small but it can also be due to the high velocity of the insect rather than the small distance. To investigate this hypothesis, Judd and Collett (1998) trained wood ants with an upright and an inverted cone. The experiments showed that ants tend to perform the turn-back-and-look approach as honeybees and wasps perform it. Upon leaving the nest, ants performed a tortuous path, turning back and looking towards the landmark. As the ants move further away, the turn-back-and-look approach is diminishing, revealing that the ants sample the landmark at a higher rate when close to it, whereas further away the sampling rate drops as the image size transforms gradually.

### 2.4.1 Path Integration

Path integration[†] in insects, refers to the ability to produce a home vector having integrated angles and distances during the outward journey (Collett et al., 1999). This home vector can produce a straight route back to the nest. In other words, although a foraging trip might not be a route viewed as a straight line, the return trip is completely optimised and previously covered route needs not be retraced. This is supported by experiments conducted by Wehner et al. (1996) and reporting that if an ant is caught on its way home and released at another site, the ant still follows its stored home vector. In other words, it traverses a path parallel and equal to the home vector, though the release site might be completely different from the site that the ant was caught. Figure 2.3 shows a schematic drawing of how an ant performs path integration during a meandering foraging journey. Ants can cover a tortuous path up to 250 m while honeybees can cover a distance of 10 km or 23 km for the *Orchida* bee.

For orientation, the ant *Cataglyphis* uses the polarisation pattern of the sky. This is achieved by its visual system, at the upper part of its facet eyes (Wehner, 2001). There, exist polarisation detectors which detect the various patterns created by the scattering of sun light in the atmosphere. For orientation of honeybees, Collett and Baron (1994) argue that this is achieved by the recognition of magnetic fields. For information on distance travelled, there are two theories; one pertaining to energy consumption while the other is based on visual cues. In honeybees, there is a strong evidence that this is achieved by means of retinal image motion (Srinivasan et al., 1996, 2000) while for the ant *Cataglyphis fortis* Ronacher et al. (2000) support this is caused by idiothetic means.

In a recent work by Wittlinger et al. (2007) experiments showed that ants employ a kind of 'step counter' for path integration. They trained the desert ant *Cataglyphis fortis* to walk a distance of 10 m in linear alloy channel from their nest entrance to a feeder. They

[†] Path integration is a term restricted to biology. The general term is *dead reckoning*, taken from nautical navigation.

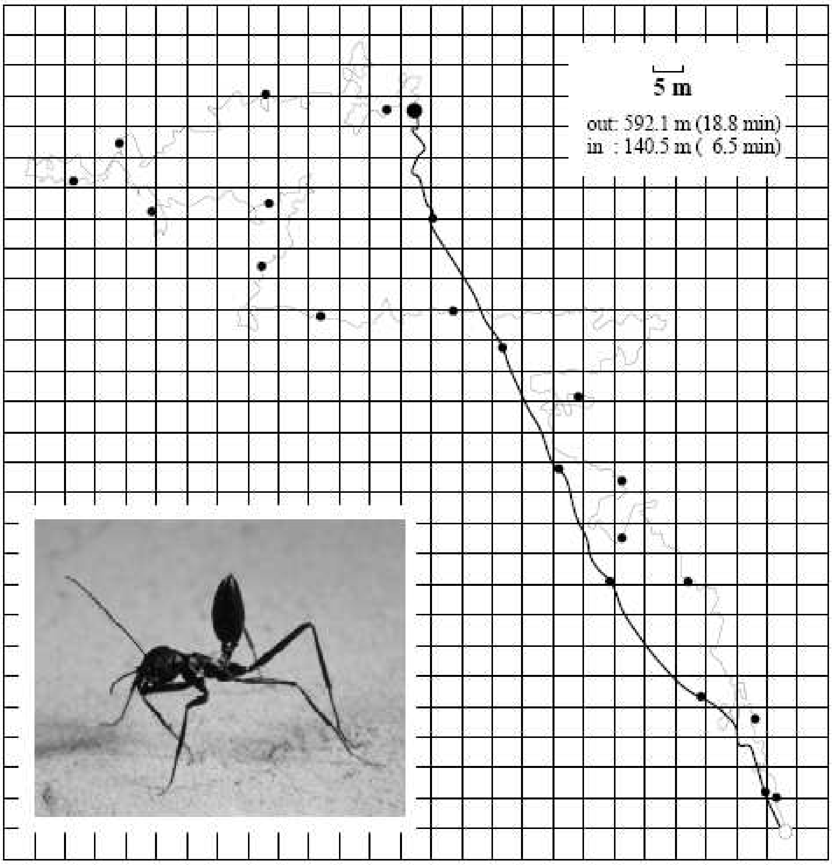


Figure 2.3: Path integration in ants. The tortuous path depicted in grey-colour line represents the foraging trip of the ant. The almost straight black line represents the homeward journey of the ant after having used its path integration system. On the top right corner the distance and time is given for both the foraging and return trip. As it can be seen, path integration in ants reduces significantly the amount of time to return from a distant and spiral-like foraging journey. Figure from Lambrinos et al. (2000).

then extended the legs of the ants by placing stilts. The tests showed that ants misgauge travel distance to feeder by overestimating the distance to it. The inverse occurred when trained ants had their legs shortened ('stumps'). The ants underestimated the travel distance to the feeder. The experiments revealed that ants used the same number of strides both in training and in tests even though their legs had been elongated or shortened. The same number of steps used by the ants shows that they use a kind of *pedometer* that counts the number of steps they have walked. Nevertheless, Collett et al. (2006) argue that ants cannot measure distances simply by counting steps. They base their argument on the fact that each stride counted by insects must be tagged with a length. An ant may start its journey by walking slowly, then it may run faster when it carries a piece of food and walk very slow when it carries a large item. Thus, they conclude that the *pedometer* is not a possible explanation as ants adjust their speed by changing both stride length and stride frequency.

In some other experiments, Grah et al. (2005) trained the desert ants *Cataglyphis fortis* to visit a feeder while travelling on undulating terrain. Their experiments showed that the ants used the ground distance as opposed to walking distance to feed into their path integrator. The ants, thus, updated their home vector using only the horizontal plane. Path integration is a prominent method used by insects, but is not the only one. The reason for this is that path integration is inherently erroneous method both in ants and in robots. Errors are accumulated especially when a wind blows and an insect can deviate from its route. This is one of the reasons why path integration is not the only strategy employed by insects. Visual landmarks play an important role in the navigation, and in particular homing, of insects.

### 2.4.2 Visual Landmarks

It has been shown that ants and honeybees use visual landmarks to orient and direct themselves towards a goal position. When other strategies such as path integration have failed, ants resort to navigation by landmarks (Wehner and Raber, 1979). Honeybees, too, select landmarks for visual homing that are close to their hive (Cheng et al., 1987). They distinguish between near and far landmarks by the retinal size and position of the landmarks. Extensive experiments conducted by Collett (1996); Graham and Collett (2002); Pratt et al. (2001) have shown that ants follow large prominent landmark lying across or near the route to the goal and that they perform a route parallel to an extended landmark such as a wall. Their distance from the extended landmark is dependent on the height of the landmark. The larger the height of the landmark, the larger the distance ants have against that landmark. By continuously following the height of the landmark, ants guide themselves to the goal. The height of the landmark, as it was previously mentioned, is calculated by its position in the retina.

In a recent paper by Dacke and Srinivasan (2008) the counting ability of honeybees is investigated. In particular, Dacke and Srinivasan (2008) trained honeybees to find a food source in a 4m long tunnel by placing landmarks in it. During training the position of the landmarks varied as well as the position of the rewarded landmark which was set to be between 120cm to 320cm away from the tunnel entrance. In addition, the shape and size of landmarks varied from experiment to experiment. The results showed that the bees hold a sequential counting ability of landmarks. Although the distance to the food reward varied in order for bees not to be able to identify the position of the landmark from its distance, bees were able to reach the food reward provided the landmarks passed en route did not exceed the number four. The number of landmarks that contained the food reward was the same both in the training and in the test, but was changed every time a new training was taking place.

### 2.4.3 Optical Flow

A large number of insects use optic flow for navigation. Optic flow in insects like *Drosophila* use the apparent visual motion of objects to supply information about the three-dimensional structure of the environment. The fly *Drosophila* uses optic flow to pick near targets. Collett (2002) shows that the task of evaluating distances between objects is made easier by making side-to-side movements of the head strictly translational and disregarding any rotational components that can influence the distance to the objects. Looming can also distort the actual distance to the object as the apparent size compared to the physical size of the object differs. In his experiments (Collett, 2002) ascertains that *Drosophila* like many insects limit rotational flow during exploratory locomotion. In fact, *Drosophila* move in straight-line segments and restrict any rotation to saccades at the end of each segment. Schuster et al. (2002) have used virtual reality techniques to show that fruit flies use translational motion for picking up the nearest object while disregarding looming (image expansion).

Ladybirds also move in straight-line segments and rely on translational optic flow rather that looming cues. Other animals like locusts and mantids turn their head from one side to the other just before jumping. Kral and Poteser (1997) suggest that locusts and mantids use translation motion to infer the three-dimensional structure of the environment and in particular the distance to the object they wish to approach. In some other experiments performed by Tautz et al. (2004) trained bees had to travel large distances across various scenes that included both land and water. The results showed that the flights over water had a significantly flatter slope than the one above land. That suggests that the perception of the distance covered by bees is not absolute but scene-dependent where the optic flow perceived is evidently larger. This may suggest why some bees are drowning by 'diving' into lakes or the sea while flying above water. The distance and direction to a food source is communicated in the bees by means of waggle dances that integrate retinal image flow along the flight path (Esch et al., 2001; von Frisch, 1993).

Esch and Burns (1996) performed experiments with trained bees to travel between two tall buildings. The bees that had flown at a higher height exhibited shorter waggle dance component than that expressed by bees travelling the same distance but at a lower height. They concluded that bees use optic flow to gauge distances to targets. The same conclusion comes from the work of Srinivasan et al. (1996) that trained bees to fly through a 20 cm wide tunnel until they reach a food source. The tunnel had textured walls in order to provide high optical flow signals to the bees. When the bees tested to fly in a narrower tunnel, they searched for the food source at a shorter distance than what had been trained. Inversely, when the bees tested to fly in a wider tunnel, the distance they covered until searching for the food source was longer than the distance they had been trained. The optic flow per unit of distance experienced by bees was higher in the narrow tunnel and lower in the wide tunnel. Therefore, Srinivasan et al. (1996) concluded, too, that bees integrate over time the amount of optic flow they experience and that the target location is a summation of the total amount of flow.

However, as stated by Collett et al. (2006) it remains a mystery how bees control height because bees tend to keep their moment-to-moment optic flow signal constant during flight by increasing speed when they gain height and reducing speed when they approach surface. Collett et al. (2006) pose the question whether bees employ static cues, such as texture grain size to control height. The findings of this thesis may explain that bees use optical flow of landmarks to localise.

One more parameter which has been tested extensively is the relationship between colour vision and motion. von Frisch (1993) and Chittka et al. (2001) report that although bees have excellent trichromatic colour vision that they use, for example, for the identification of flowers, colour is not related to motion (Zhang and Srinivasan, 1993). Experiments conducted by Chittka and Tautz (2003) report that even though distance estimation is colour blind there is a strong correlation between green contrast and waggle dance probability while there is no correlation between any other spectral parameters and waggle dance probability. That, therefore, indicates that the green receptor signal is used for distance estimation in honeybees.

### 2.4.4 Searching

Wehner and Srinivasan (1981); Müller and Wehner (1994) demonstrate that the desert ant (*Cataglyphis*) does not perform a random walk but rather it resorts to a piloting or a systematic search, a spiral pattern-like behaviour, when all other information has failed. For example, when an ant cannot locate its nest after returning from a foraging trip, because of a failure of its path integration system, it resorts to a searching technique. A fail in the path integration system can be caused, for example, by a cross wind, making the home vector not to coincide with the nest entrance. In particular, this search pattern consists of a system of loops of ever-increasing size centred about the origin. Computer simulations performed by Müller and Wehner (1994) have shown that searching strategy adapts to the probability density of finding the target, i.e., the nest. In other words, these patterns reveal that the target location is most heavily searched, indicating that the behaviour of the ant *Cataglyphis* is geared towards finding the nest in an optimal way.

## 2.5 Robot Strategies

This section describes robot strategies, in particular, the turn-back-and-look behaviour which has been implemented on various autonomous agents. But, what does *agent* exactly mean? The term *agent* is somewhat ambiguous, and in this context is considered

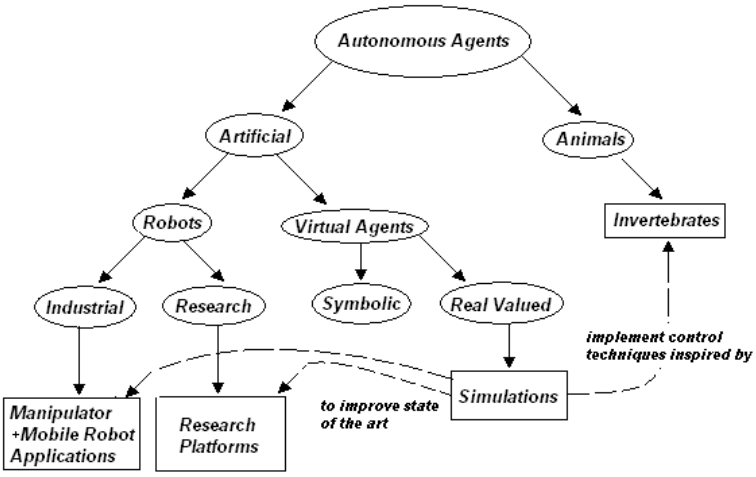


Figure 2.4: Partial agent taxonomy. Figure adapted from Sunderland (2006).

to mean a system that operates independently, contained within a shell, or 'body', which mediates its interactions with the outside world (Sunderland, 2006). What constitutes the body is context dependant. In other words, when we are referring to animals or robots, body implies the collective term of its material components, whereas in software agents it is the methods through which they can sense and modify their environment. Figure 2.4 shows a partial representation of agent taxonomy. The work presented in this thesis involves the implementation of real-valued virtual agents that are inspired mainly by invertebrates for use in both industrial and research robots. The difference between the real-valued and symbolic virtual agents lies in the fact that the latter use symbolic logic.

Lehrer and Bianco (2000); Lehrer (1991, 1993); Lehrer and Collett (1994), and Fry and Wehner (2005) describe the turn-back-and-look behaviour from the biological point of view. To complement this, Lehrer and Bianco (2000); Bianco and Cassinis (2000), and Thompson et al. (1999) describe the TBL behaviour from the robotics point of view. In particular, in their work TBL is used for the selection of stable landmarks. TBL is used for dynamic landmark selection, whereas for static landmark selection the 'The Valley Method' proposed by Mori et al. (1995) is used. Figure 2.5 shows a distortion matrix of a reliable landmark. In this method various landmarks of local uniqueness are selected compared to the area immediately surrounding the landmark.

Equation 2.1 is a boolean measure to select a template where $\delta$ is a given threshold, $g'$ is the (local) minimum value found in a circle given a radius around $g$, that is the global minimum in the correlation matrix. Figure 2.5 depicts the process of a correlation matrix where the global minimum, that is $g$, is found at the centre of the grey area whereas the local minimum, that is $g'$, is found in the grey area.

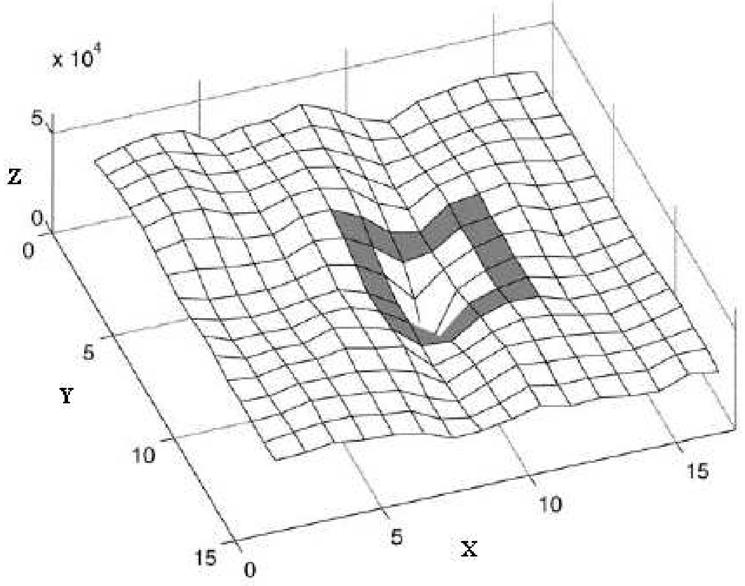


Figure 2.5: *The Valley Method.* The uniqueness of the landmark is depicted by the sharp point in the centre of the image, i.e., correlation, being represented by the *z-axis*. The *x-axis* and *y-axis* represent the columns and the rows of the matrix, respectively. Grey rectangles denote the pixels surrounding the landmark. Global minimum is located at the centre of the grey area whereas local minimum is found in the grey area. Figure from Bianco and Cassinis (2000).

$$r = 1 - \frac{g}{g'} > \delta \tag{2.1}$$

By relaxing the 'The Valley Method' the following eqn. 2.2 is obtained that calculates the reliability of each landmark, i.e., how stable it is through time where $r$ is the measure of uniqueness of the landmark. The greater the value of $r$, the more unique a landmark is compared to its neighbourhood (grey squares).

$$r = 1 - \frac{g}{g'} \tag{2.2}$$

The reliability of the $r_l$ of the landmark $l$ is continuously evaluated and updated using the following eqn. 2.3,

$$r_l = \frac{\sum_{i=1}^{n} r_l^i}{n} \tag{2.3}$$

where $n$ is the total number of steps taken during the TBL until that time, and $r_l^i$ is the reliability of landmark $l$ calculated at time $i$.

## 2.6 Strategies of Navigation

In navigation, three questions that must be answered by an autonomous agent each time, these are; *where am I*, *where am I going*, and *how will I get there*, though the most important for the homing task is the last one (Franz and Mallot, 2000). However, based on the sensory information available, the navigation task, and the state of the agent, different strategies can be followed. Redish (1999) presents five strategies an agent can take, these are:

I. Random navigation

   When there is no information about the goal, the agent moves around randomly in the environment until it finds the goal otherwise it switches to another strategy.

II. Taxon navigation

   Taxon navigation, as mentioned in the previous section, refers to attraction of the agent directed by a stimulus, e.g., landmark or beacon.

III. Praxic navigation

   Praxic navigation comes from the term *praxis* meaning ‘action’. The agent can execute a constant motor programme. A special case of praxic navigation is path integration.

IV. Route navigation

   Route navigation is the sequence of taxon and praxic navigation strategies dependent on the sensory input.

V. Locale navigation

   Locale navigation refers to the learning of location of the goal relative to a constellation of cues. Redish (1999) claims that locale navigation depends on the local view, path integrator coordinates, head direction, place code, and goal memory.

Franz and Mallot (2000) present a hierarchy of two strategies, namely local navigation and way-finding. The former refers to the navigation process that requires the recognition of only one location, the goal location, while the latter one involves the recognition of several places.

> *“Way-finding relies on local navigation skills to move from one place to another, but it allows the agent to find places that could not be found by local navigation alone. The distinction between local navigation and way-finding is not necessarily associated with the scale of the environment in which the agent moves: By simply following a locally measured compass direction, for instance, one can find places that are thousands of*

> *kilometres away, whereas finding a particular room in a campus building already requires considerable way-finding skills"* (Franz and Mallot, 2000).

This navigation scheme is a modification and extension of the scheme proposed by Trullier et al. (1997). The two main divisions of navigation and their subdivisions presented by Franz and Mallot (2000) are:

I. Local navigation

   A. Search

   B. Direction-following

   C. Aiming

   D. Guidance

II. Way-finding

   A. Recognition-triggered response

   B. Topological navigation

   C. Survey navigation

Search forms the simplest case of navigation. This can be the case when the agent performs a systematic or random search in pursuit of the goal. It requires the simplest form of locomotion and goal detection. In addition, search requires a large amount of time compared to other strategies. Direction-following may be extracted either from allothetic (based on an external reference) information sources such as the magnetic field of earth or pheromone trails, or they can be extracted based on idiothetic (internal reference) information such as proprioceptive signals or an inertial compass. Direction-following is more effective than search but, it allows the agent to find its goal by following trails (e.g., pheromone). Examples include path integration or trail following.

Aiming requires the agent to orient its body axis towards a salient cue that is perceptible during the approach, such as olfactory, auditory, or other. Guidance is the process whereby a certain egocentric relationship with respect to the landmark is maintained. When a goal is not marked by a salient cue, the agent can be guided by the spatial configuration of the surrounding objects. Visual homing is an example of guidance. A nautical example of guidance would be a ship that tries to reach a fixed position between several islands. These four methods fall under the local navigation since they lead to a single location with the help of locally available information.

Recognition-triggered responses connect two locations by means of a local navigation method. They require the recognition of the goal as well as of the starting location. The recognition of the staring location triggers the activation of a local navigation method leading to a goal. *Routes* are sequences of recognition-triggered responses in which the achievement of a place triggers the reach of the next one. Topological navigation requires the agent to have a representation of places and their connections. It integrates route navigation and allows for finding alternate routes to a goal in the presence, for example, of obstacles. However, such a process requires that the agent has previously visited the route. These routes can be expressed mathematically in the form of graphs, where vertices represent places and edges represent local navigation methods. As a result, by building graphs alternative routes can be generated, but cannot be novel and efficient if the routes have not been visited previously.

Lastly, survey navigation forms the most complex method of navigation since it requires the embedding of all known places and their spatial relations into one frame of reference. Apart from topological information the agent can extract metric information that can be used for the finding of novel routes or shortcuts between unconnected routes, or detours around obstacles in unknown terrain. These last three methods fall under the way-finding spatial behaviour. This work is trying to use both local and way-finding methods for navigating an agent in unknown environments. The next section discusses the main biological models and their implementation to robots.

## 2.7 Models of Navigation

What constitutes a good model? A biological model can vary from the number of legs an animal possesses to a detailed description of its neural circuitry mechanisms. Webb (2001) discusses what constitutes a good model and how biological behaviour should be modelled. As previously stated, visual homing falls under the *guidance* form of navigation. Visual homing with the use of landmarks is employed by social insects when, after foraging, ants return home. Nonetheless, this is not always the case. Homing is also considered when the goal is at the food source and not necessarily at the nest or the hive, where an insect need to return back. In addition, homing is used in conjunction with path integration (the integration of distance and direction) and systematic search (e.g., spiral patterns of motion), though landmark navigation can be used after the afore-mentioned approaches have failed (Wehner and Raber, 1979) or when an insect is returning to the goal location having previously stored a snapshot of the environment and the goal location.

Most of the models used for robotic homing fall under the so called local homing strategy (Hong et al., 1991). Local homing refers to one only image taken at the goal position. The model then compares that picture with a stored one to infer the homing vector. However, there exists another approach, the so called associative homing (Nelson, 1991; Jogan and Leonardis, 2000), where it samples the environment at a number of different locations. The disadvantage of the associative homing lies in the storage and processing requirements of images. The larger the sampling of the environment, the greater the number of images to be processed. Yet, it is not quite clear whether insects sample their environment a number of times. Collett (1996) argues the association of scenes and vectors. For example, when an insect arrives at a goal location it associates the home vector acquired by its path integration system with the local scene. This hypothesis has been supported by experiments conducted with bees (Menzel et al., 1998). Wehner et al. (1996) juxtapose, however, by arguing that insects acquire images in memory in series which are retrieved in order. For example, a landmark or image that has been stored on insect's memory will retrieve the next image stored that will lead the insect to the next location. Thus, achieving visual homing.

### 2.7.1 The Template Hypothesis

The template hypothesis was first conceived by Cartwright and Collett (1983, 1987b) who suggested it as a model for visual homing of honeybees. In their experiment, they trained bees to enter a room from a window few metres away from the hive of the bees. The room was prepared to eliminate visual cues. Inside the room, was located a food source surrounded by some artificial landmarks. These were black cylinders of 40 cm in height and 4 cm in width. Cartwright and Collett (1983, 1987b) found that bees learnt the retinal size and bearings of the landmarks as viewed from the food source. This revealed that honeybees stored a snapshot of the goal position. When the array of landmarks was modified or when food source was removed, the bees started searching where the apparent size and bearings of landmarks was matching the most the snapshot image. This also revealed that there was a matching processing between the snapshot image taken at the goal position and the snapshot taken at the current position of the honeybee.

A similar experiment was conducted with the ant *Cataglyphis bicolor*. The experiment by Wehner and Raber (1979) shows that ants try to match the current retinal image with a stored one. Their experiment involved training ants in a location between two landmarks. When the landmarks moved apart twice the original distance, then the ants gathered around each landmark. When, on the other hand, the landmarks moved apart again twice the distance, but had a double size this time, the ants gathered between the landmarks. Again, this experiment supports the template hypothesis. Figure 2.6 shows experimental data collected by Wehner and Raber (1979) and simulation results from Moller (2001).

### 2.7.2 The Parameter Hypothesis

The parameter hypothesis was first conceived by Anderson (1977) who conducted experiments in honeybees, too. The experiment included eight landmarks forming a circle, in

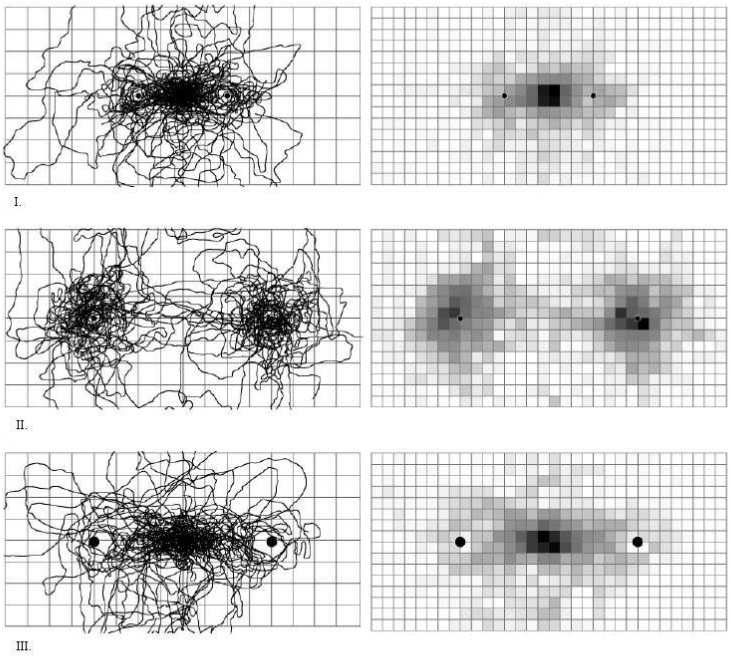


Figure 2.6: Experimental and simulation data showing the behaviour of ants according to the size of landmarks stored on the retina. Landmarks are depicted by black circles. **I.** Nest in the centre of the landmarks. **II.** Double distance of landmarks from the nest with fixed their size. **III.** Double landmark size and distance. Figure from Moller (2001).

the centre of which there was a food source. Three of the landmarks were removed and the tests showed that the bees did not gather in the position where they could achieve a best retinal match and bearing of the landmarks, but instead, they moved towards the inside part of the circle. Anderson (1977) suggested that honeybees searched to restore a measure of *surroundedness*.

The following section presents the major models of homing based on the template and parameter hypotheses. Vardy (2005) provides a review on the approaches and a classification based on the hypothesis used. The majority of the models discussed below, however, adopt the template approach. It should be noted that the following models make use of panoramic cameras as they resemble better, but are not identical, to the visual system of insects, the *ommatidia* (single eyes that make up the compound eye). Thus, the images captured by such cameras cover a 360° field of view. Panoramic cameras are simple in its conception, i.e., they consist of a camera turned upwards and towards a parabolic mirror that reflects the rays of the environment.

### 2.7.3 The Snapshot Model

The *snapshot* model is an implementation of the template hypothesis. It requires a panoramic snapshot of the goal position, be it a hive, nest, or a food source. Along with the snapshot the compass direction is stored. The snapshot model is an image matching process between a snapshot taken at a goal position and a snapshot containing the current view. The image obtained from the omnidirectional camera is unwrapped and a threshold operation is performed to yield a one-dimensional black and white image. The landmarks are denoted as black marks on the image. Then, this is compared with the snapshot of the current view to produce the homing vector. The *homing vector* is a two-dimensional vector pointing towards the home position, and is obtained by summing up all radial and tangential vector components. A *radial vector* is directed towards the landmark if it has grown smaller in the current snapshot or away from the centre of the landmark if it has grown larger. The *tangential vector* is attached tangentially to the landmark to denote the direction of the landmarks in the snapshot image. All these vectors are either unit or zero vectors.

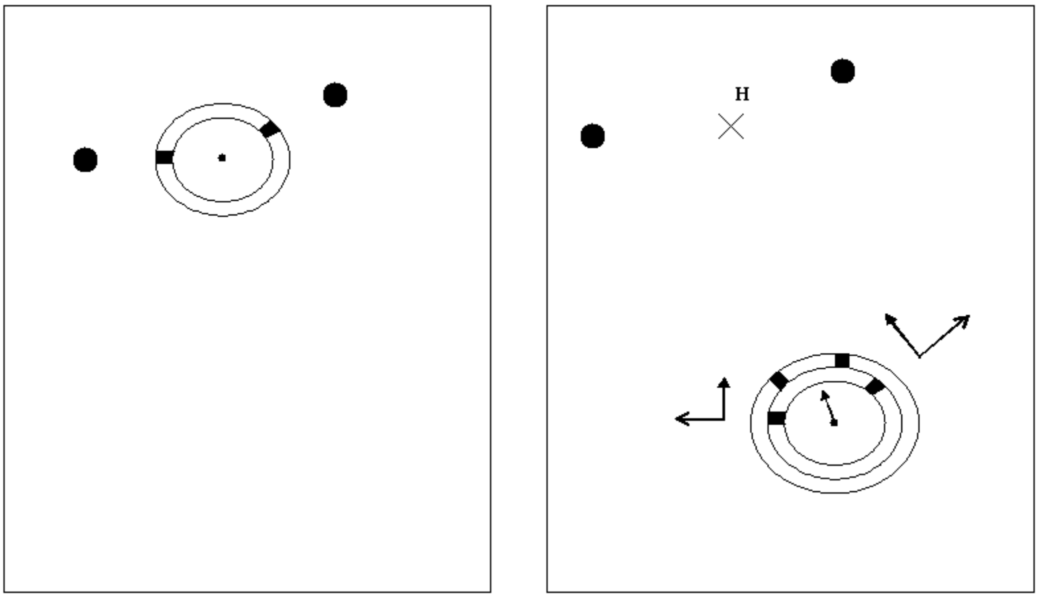
The environment in which this model has been tested is quite simple with some cylinders or other geometrical objects acting as landmarks, thus making the image processing a very easy task. Figures 2.7(a) and 2.7(b) show a representation of the model. In Fig. 2.7(a) the insect is at the goal location. The two black circles denote the landmarks and the two black squares represent the landmarks as they have been imprinted in the panoramic image. In the adjacent figure, Fig. 2.7(b), the insect has been displaced and the inner circle represents the snapshot image while the outer circle represents the current image. The filled head arrows depict the *tangential vectors*, $v_t^i$, while the open ones the *radial vectors*, $v_r^i$. The arrow in the centre of the circles denotes the home vector, $v_h$ pointing towards the home, **H**, location after having summed up the *tangential* and *radial* vector components. The mathematical formulation of the snapshot model can be expressed with the following eqn. 2.4,

$$v_h = \frac{\sum_i (3v_r^i + v_t^i)}{||\sum_i (3v_r^i + v_t^i)||}. \tag{2.4}$$

### 2.7.4 The ALV Model

The *Average Landmark Vector* (ALV) model (Lambrinos et al., 2000) uses, too, a processed panoramic image but, in contrast to the snapshot model, it need not be stored. Only a two-dimensional vector for each landmark needs to be stored that points to the direction of the landmark. Matching and unwrapping of the image are not required, since the calculations are performed on the basis of vector components. Thus, ALV is more parsimonious than the snapshot model. Nevertheless, snapshots in the ALV model have to be captured and processed to produce a one-dimensional picture, as in the snapshot model.

Figures 2.8(a) and 2.8(b) depict the Average Landmark Vector (ALV) model. In Fig. 2.8(a) the insect is at the home location where two vectors, $l_1^i$, point to the centres of the



(a) Snapshot at goal location. Landmarks are represented by black circles. The black rectangles denote the landmarks in the one-dimensional, black and white, panoramic image.

(b) Snapshot at current location. The inner circle represents the image at goal location while the outer circle the image at the current location. Home position is denoted as H. Filled head arrows denote the tangential vectors while the open ones the radial vectors. The homing vector pointing towards the home position is depicted with the arrow in the centre of the circle.

FIGURE 2.7: Snapshot model at goal location 2.7(a) and at the current location 2.7(b).

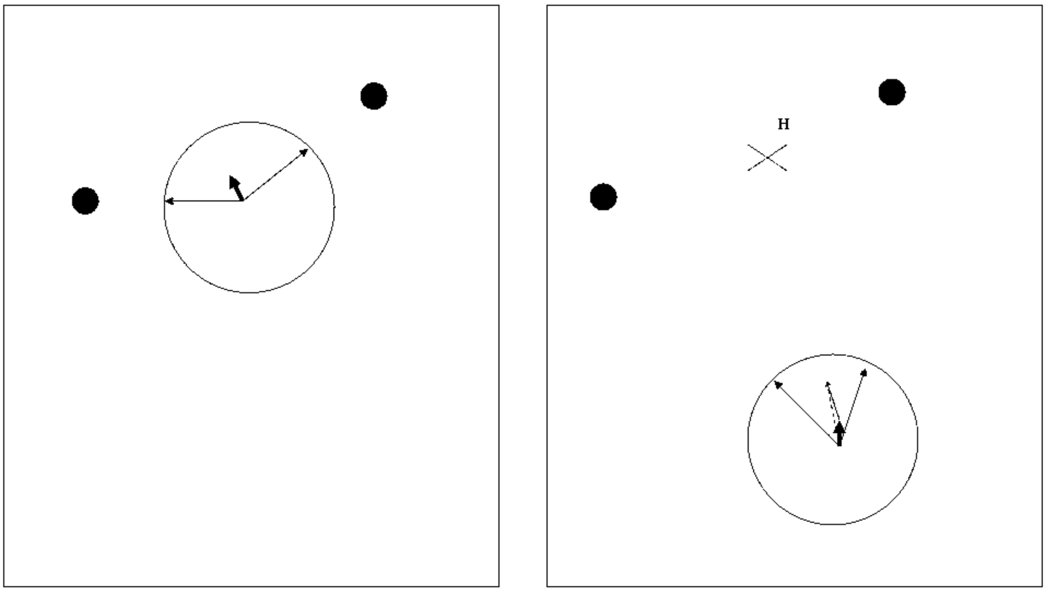
corresponding landmarks $n_1$. The bold arrow represents the average vector $ALV_S$. In the adjacent figure, Fig. 2.8(b), the agent (robot, in this case) is at a displaced position, having again two vectors, $l_2^i$, pointing in the direction of the centres of the landmarks, $n_2$. The home vector (the arrow with the unfilled head), $v_h$, is calculated by subtracting the average vector (the short, bold arrow) at home location, $ALV_S$, from the average vector (the dashed line), $ALV_C$, at the current location. In Fig. 2.8(b), **H** denotes the home location. The mathematical formulation of the ALV model is expressed with eqn. 2.5,

$$v_h = \frac{1}{n_1}\sum_{i=1}^{n_1} l_1^i - \frac{1}{n_2}\sum_{i=1}^{n_2} l_2^i \tag{2.5}$$

where $ALV_C$, eqn. 2.6, is the current viewed snapshot,

$$ALV_C = \frac{1}{n_1}\sum_{i=1}^{n_1} l_1^i \tag{2.6}$$

and $ALV_S$, eqn. 2.7, is the snapshot at goal location,



(a) Snapshot at goal location. Landmarks are represented by black circles. The two long arrows represent the vectors that point to the direction of the landmarks. In ALV model the image is not stored, but rather the vectors pointing to landmarks. The bold arrow represents the average vector.

(b) Snapshot at current location. The two long arrows represent the vectors that point to the direction of the landmarks. The bold arrow represents the average vector at home location, denoted as H. The dashed arrow represents the average vector at current location while the arrow with the unfilled head represents the home vector. The home vector is calculated by subtracting the average vector at home location from the average vector at current location.

FIGURE 2.8: Average Landmark Vector (ALV) model at goal location 2.8(a) and at current view 2.8(b).

$$ALV_S = \frac{1}{n_2} \sum_{i=1}^{n_2} l_2^i \tag{2.7}$$

This reduces to eqn. 2.8,

$$v_h = ALV_C - ALV_S \tag{2.8}$$

Two mobile robots, *Sahabot I* and *Sahabot II*, were built by Lambrinos et al. (1997, 2000); Möller et al. (1998) in order to test and simulate the behaviour of the ant *Cataglyphis*. The robots model the polarisation pattern of the sky that *Cataglyphis* uses for path integration. In the robots, this is implemented with photodiodes covered with a polarisation filter and a blue filter providing sensory information. Both robots make use of panoramic cameras, i.e., a camera pointing upwards, towards a conical mirror. However, it has to be noted that such an approach does not model faithfully the vision of ants. In *Cataglyphis* the height of the eyes from the ground is about 0.5 cm whereas in the panoramic cameras used, the system's distance from the ground was about 25 cm. The path integration system was implemented on the robots with wheel encoders. The formula for representing the position and location of the robot according to the sensory information is given by the following formulas 2.9 and 2.10,

$$x(t + \Delta t) = x(t) + \cos(\theta(t))s(t)\Delta t. \tag{2.9}$$

$$y(t + \Delta t) = y(t) + \sin(\theta(t))s(t)\Delta t. \tag{2.10}$$

where $\Delta t$ is the time step, $(x(t), y(t))$ is the position of the agent at time $t$, $\theta(t)$ is the heading angle at time $t$, and $s(t)$ the speed at time $t$. Sahabot II implements also the snapshot behaviour of the *Cataglyphis*. The experiments with both robots took place in the Tunisian desert having as landmarks black cylinders. The image processing was once again limited to a one-dimensional, binary picture.

The *snapshot* and the *Average Landmark Vector* are two models that have been inspired by the way insects perform homing. On one hand, their main advantage is the simplicity of the method that entails a low computational complexity. On the other hand, their disadvantages are that both methods are applied at the end of the homing process, that is, when an agent is close to its home position. Moreover, all landmarks need to be visible both in the current snapshot as well as in the stored snapshot, that is, they must be the same landmarks in both snapshots.

## 2.8 Various Models of Navigation

This section discusses an assortment of other homing models that have been developed, most of which resemble or adopt characteristics from either the snapshot or the ALV model of homing.

### 2.8.1 Analog ALV Model

Möller (2000) presents the analog variant of the ALV model. Since the ALV model is a parsimonious model it is easier to implement in analog hardware. Analog hardware shares a number of information processing principles with its biological counterparts. The analog robot built consisted of 32 photo-diodes arranged horizontally with a standard operational amplifier circuit for amplification comprising the visual sensor for edge extraction. Two flux gate magnetic field sensors arranged orthogonally act as a compass to rotate the ALV model. The ALV vector is stored in two capacitors and then is applied to the motors by rotating the right wheel according to the $x$ component of the vector, and applying the $y$ component of the vector to the left-hand side wheel. The analog robot built provided a test bed for the biological hypotheses, i.e., how various neural mechanisms take place in the insect's brain, thus, providing an insect with visual homing capabilities. The work by Möller (2000) suggests that ALV can be implemented in analog hardware mainly because of its simplicity and the little computation costs it has. Furthermore, it suggests that in some landmark setups, the behaviour of the analog model resembles that of the bees.

### 2.8.2 Warping Model

Franz et al. (1998a,b) adopt a different approach compared to that of searching correspondences. The model makes use of panoramic images reduced to one-dimensional images with greyscale values. Their model is based on image warping, searching for possible directions, orientations, and distances. The image captured is warped and compared with the image taken at the goal position. A number of different possible vectors are compared and the best match is chosen. The model was tested in a miniature world, consisting of miniature toys. It achieved good results, but its biological plausibility has been questioned. A variant of the warping model has been implemented by Stürzl and Mallot (2002) which uses range images rather than intensity images. The model performs well in illumination changes, but is limited to 1 meter distance inference mainly due to the small baseline used.

## 2.9 Navigation and Homing using SLAM

This section discusses the Simultaneous Localisation and Mapping (SLAM) problem for navigating mobile robots. SLAM has risen to one of the most research-intensive problems in the robotics field. SLAM is not used as much for homing as for navigation, in general. In this approach, when a robot navigates it needs to acquire its location in order to build an accurate map. However, in order for an accurate map to be constructed, accurate localisation must have been inferred. Therefore, localisation and mapping are two inter-dependent tasks. Localisation and mapping can thus be considered to be a “a chicken and egg” problem, requiring both accurate position estimates of the robot and of the surroundings. In any SLAM algorithm the number and location of landmarks is not known a *priori*. Landmark locations must be inferred from observations obtained by sensors.

Smith et al. (1988) used an Extended Kalman Filter (EKF), instead of the simple linear Kalman filter, to solve the SLAM problem. The EKF uses linearised kinematic and observation equations for generating state predictions. They modelled the robot pose and the map parameters in the state vector. The readings from the sensors were used to update the map parameters and the pose of the robot relative to a global frame of reference. The disadvantage of the approach, however, is that the increased number of features added to the map expands the matrix that keeps track of all the features and thus, adding to the increased computational costs of the approach. There have been attempts, however, to reduce the unnecessary features tracked from the state vector (Dissanayake et al., 2000).

FastSLAM (Hahnel et al., 2003; Montemerlo et al., 2002) is another method that integrates particle filters and Extended Kalman Filters. FastSLAM tries to alleviate the problem of *data association*, that is, the problem where landmarks look alike which is prevalent in the previous methods. Most of the above mentioned methods use mainly odometry information and laser sensors to support the SLAM methodology. Nonetheless, there has also been a SLAM-based method that employs a single camera (Davison et al., 2007) to infer the 3D trajectory of a monocular camera in an unknown environment. This vision-based SLAM has been widely known for years as the 'structure from motion' problem and it had been researched in parallel and in ignorance of the robotics community. Other sensors which have been used for localisation include Radio Frequency Identification (RFID) technology. For example Hahnel et al. (2004), proposes a model based on RFID technology and a laser-based FastSLAM approach to effectively determine the location of RFID tags.

Amigoni et al. (2004) proposed a method for map building without using any odometry information. Their method builds a geometrical global map based on various scans of the environment taken at different instants of time. They try to integrate the maps using three different methods, namely sequential, tree, and pivot methods. However, they do not keep track of the position of the manually driven robot. An approach which uses Dynamic Programming (DP) for the real-time self-localisation of a robot is explained by Einsele (1997). In this work a panorama laser range finder (PLRF) is employed. Similarly to the previous work, a matching between present preprocessed scans and already stored scan data is taking place. The task of preprocessing method is to extract line segments from the acquired range data. Moreover, in their approach they make use of local coordinate systems linked together by topological information. No global environmental map is built, but they rather focus on local 'distinctive' places to build local maps for self-localisation.

One of the fundamental problems in the SLAM approach is the accumulation of error as the robot navigates in an environment. The harsher the environment, the greater the likelihood for accumulation of errors due to noise in sensor readings. One such error is caused by the dead reckoning system of the robot. The slippage of the terrain causes erroneous sensor readings. Though most of the approaches make use of range sensors such as laser and sonar, a work by Davison and Murray (2002) approaches the problem of SLAM with active vision sensors. Davison and Murray (2002) used SIFT features that are invariant to scale and rotation to estimate their position in the three-dimensional space. Furthermore, sparse prior knowledge is incorporated into the maps to reduce the uncertainty in the global frame of reference. In a subsequent work, Davison (2003) uses a real-time approach to the localisation and mapping problem using a single camera. The depth of features is inferred by calculating inverse depth.

### 2.9.1 Mathematical Formulation of the SLAM Problem

In this section the mathematical formulation of the SLAM problem is described, and in particular the probabilistic approaches used to estimate the position of the vehicle and the location of the landmarks within the environment. The SLAM process consists of a number of different steps; such as landmark extraction, data association, state estimation, state update and landmark update. There are various ways of solving each of the different steps and a number of different sensors for providing the input. The most common sensors used for SLAM, however, are laser scanners and odometry sensors. In lieu of the laser scanners, sonar or vision sensors can be used but each one of them has its advantages and disadvantages. Laser scanners are reliable and apart from from their suitability in landmark extraction, they can also be used for obstacle avoidance. Especially for the latter, laser scanners outweigh sonar and infrared sensors.

When a robot manoeuvres in an environment, it is making observations using its sensors, for example laser scanner, and is estimating its position based on the input from the odometry sensors. At time instant $k$, the following quantities are defined as described by Durrant-Whyte and Bailey (2006):

I. $\mathbf{x}_k$: the state vector describing the location and orientation of the robot

II. $\mathbf{u}_k$: the control vector, applied at time $k-1$ to drive the robot to a state $\mathbf{x}_k$ at time $k$

III. $\mathbf{m}_i$: a vector describing the location of the $i$th landmark whose true location is assumed time invariant

IV. $\mathbf{z}_{ik}$: an observation taken from the robot of the location of the $i$th landmark at time $k$.

In addition, the following sets are defined:

I. $\mathbf{X}_{0:k} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k\} = \{\mathbf{X}_{0:k-1}, \mathbf{x}_k\}$: the history of robot locations

II. $\mathbf{U}_{0:k} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k\} = \{\mathbf{U}_{0:k-1}, \mathbf{u}_k\}$: the history of control inputs

III. $\mathbf{m} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_n\}$: the set of all landmarks

IV. $\mathbf{Z}_{0:k} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k\} = \{\mathbf{Z}_{0:k-1}, \mathbf{z}_k\}$: the set of all landmark observations.

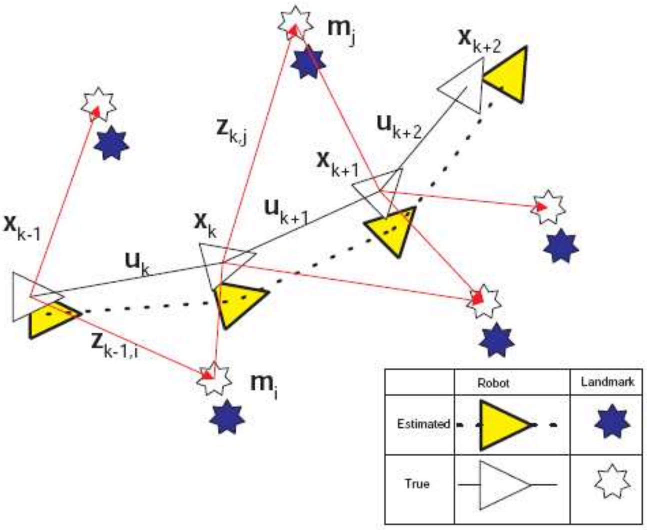


Figure 2.9: The Simultaneous Localisation and Mapping (SLAM) process that includes estimated locations of landmarks and robot based on observations taken and their true locations. Figure from Durrant-Whyte and Bailey (2006).

Figure 2.9 shows the basic idea that underpins the SLAM problem. In probabilistic form, the SLAM problem requires that the probability distribution, eqn. 2.11,

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{2.11}$$

be computed for all times $k$. The *observation model* describes the probability of making an observation $\mathbf{z}_k$ when the robot position and landmark location are known and is described in the following eqn. 2.12,

$$P(\mathbf{z}_k|\mathbf{x}_k, \mathbf{m}) \tag{2.12}$$

The *motion model* for the robot is described with the following eqn. 2.13,

$$P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \tag{2.13}$$

The SLAM algorithm is implemented in a two-step recursive (sequential) prediction time update form given by the following eqn. 2.14,

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \int P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \times P(\mathbf{x}_{k-1}, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1} \tag{2.14}$$

and the measurement update described by the eqn. 2.15,

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \frac{P(\mathbf{z}_k|\mathbf{x}_k, \mathbf{m}) P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0)}{P(\mathbf{z}_k|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \tag{2.15}$$

Equations 2.14 and 2.15 provide a recursive procedure for calculating the joint posterior $P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$ for the robot state $\mathbf{x}_k$ and map $\mathbf{m}$ at time $k$ based on all observations $\mathbf{Z}_{0:k}$ and all control inputs $\mathbf{U}_{0:k}$ up to and including time $k$.

The localisation and mapping problem can thus be split into two separate tasks and be computed independently. Therefore, the map building problem can be formulated as computing the conditional density $P(\mathbf{m}|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k}, \mathbf{U}_{0:k})$ assuming that the location of the vehicle $\mathbf{x}_k$ is known at all times. A map $\mathbf{m}$ is then constructed by fusing observations from different locations. Similarly, the localisation problem can be formulated by computing the probability distribution $P(\mathbf{x}_k|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{m})$. This time, however, it is assumed that the landmark locations are known with certainty at all times and the objective is to compute an estimate of the vehicle location with respect to these landmarks.

Most problems in SLAM accrue from errors between true and estimated landmark locations. Those errors are a result of vehicle motion, as the sensors, in particular odometry sensors, cannot give the true location of the vehicle, thus giving erroneous estimates of landmark positions. This implies that errors in landmark location estimates are highly correlated meaning that the relative location between landmarks may be known with high accuracy, even though the absolute location of landmarks is quite uncertain. Thus, the correlations between landmark estimates increase monotonically as more and more observations are made resulting in a more accurate estimate of relative location of landmarks regardless of vehicle motion. (Dissanayake et al., 2001) prove it for the linear Gaussian case.

Figure 2.9 shows that at location $\mathbf{x}_k$ the vehicle observes two landmarks $\mathbf{m}_i$ and $\mathbf{m}_j$. At location $\mathbf{x}_{k+1}$ the vehicle observes again $\mathbf{m}_j$ and two more landmarks. This allows the update model to take place and the estimated location of the robot and landmark $\mathbf{m}_j$ to be updated. This causes a chain of updates of the past observations to take place and update the landmark $\mathbf{m}_i$ as well although it is not observed at location $\mathbf{x}_{k+1}$. This occurs because the landmark location estimates are highly correlated.

#### 2.9.1.1 Kalman Filter and Extended Kalman Filter

At the heart of the SLAM problem lies the Kalman Filter (KF) (Welch and Bishop, 2006). The Kalman Filter in the SLAM problem processes data like sensor readings and actuator input and tries to estimate the position and pose of the vehicle as well as the location of landmarks. Since noise is involved in the SLAM problem, Kalman Filter assumes a zero mean uncorrelated noise that is normally distributed. The uncorrelated zero mean noise is also known as white noise. A correlated noise between time steps is normally small in robot sensors, thus, it is not considered. One more assumption made by the KF is that the process model is linear. A linear process can be the case when a robot is navigating indoors where the motion of the vehicle can be described by a linear function. For more complex cases, i.e., nonlinear, the Extended Kalman Filter (EKF) is used. In Kalman Filter the time update is described by the following eqn. 2.16,

$$\mathbf{x}_{k|k-1} = \mathbf{x}_{k-1} + \mathbf{G}_k \mathbf{u}_k \tag{2.16}$$

where $\mathbf{G}_k$ is a matrix that maps control inputs into a predicted state space (Csorba, 1997).

The observation update in the Kalman Filter is given by eqn. 2.17,

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{W}_k \mathbf{S}_k \mathbf{W}_k^T, \tag{2.17}$$

where $\mathbf{S}_k$ is defined by eqn. 2.18,

$$\mathbf{S}_k = \nabla \mathbf{h} \mathbf{P}_{k|k-1} \nabla \mathbf{h}^T + \mathbf{R}_k, \tag{2.18}$$

and $\mathbf{W}_k$ is defined by eqn. 2.19,

$$\mathbf{W}_k = \mathbf{P}_{k|k-1} \nabla \mathbf{h}^T \mathbf{S}_k^{-1} \tag{2.19}$$

The matrix $\mathbf{S}_k$ is the covariance of the innovation $\mathbf{v}_k$ which in turn is defined as additive, zero mean uncorrelated Gaussian observation errors with covariance $\mathbf{R}_k$. Innovation refers to the difference between the estimated robot position and the actual robot position. The $\nabla \mathbf{h}$ is the Jacobian of $\mathbf{h}$ that transforms representations used by sensors into state estimates. $\mathbf{W}_k$ is the Kalman weighting matrix which determines the influence of the innovation on the updated estimate (Csorba, 1997).

The Extended Kalman Filter is applied to nonlinear cases where, for example, the estimated position of the vehicle cannot be described by a linear function. This can be the case when the wheels of a vehicle are spinning without the vehicle actually moving. The time update of the EKF is given by eqn. 2.20 and eqn. 2.21,

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k) \tag{2.20}$$

$$\mathbf{P}_{xx,k|k-1} = \nabla\mathbf{f}\mathbf{P}_{xx,k-1|k-1}\nabla\mathbf{f}^T + \mathbf{Q}_k, \tag{2.21}$$

where $\mathbf{f}(\cdot)$ models vehicle kinematics and $\nabla\mathbf{f}$ is the Jacobian of $\mathbf{f}$ evaluated at the estimate $\hat{\mathbf{x}}_{k-1|k-1}$. $\mathbf{Q}_k$ is the covariance matrix of the process noise (Durrant-Whyte and Bailey, 2006).

The observation update of the EKF is given by eqn. 2.22,

$$\begin{pmatrix} \hat{\mathbf{x}}_{k|k} \\ \hat{\mathbf{m}}_k \end{pmatrix} = [\hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{m}}_{k-1}] + \mathbf{W}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{m}}_{k-1})] \tag{2.22}$$

and eqn. 2.17. The main disadvantage of both KF and EKF implementations is the computational cost of updating all landmarks and the joint covariance matrix every time a new observation is made. The computation cost, thus, increases quadratically with the number of landmarks. Another problem associated with KF and EKF is the data association. Data association refers to cases where a vehicle re-observes some landmarks and cannot realise they have already been observed. Thus, they are counted for new landmarks. This gives rise to the loop-closing problem where a robot cannot distinguish a location it has already visited.

#### 2.9.1.2 FastSLAM

The FastSLAM algorithm (Montemerlo et al., 2002) was developed to tackle the limitations of the KF and EKF solutions. As was shown earlier, the KF and EKF models suffer from the computation cost of updating all landmarks and covariance matrices every time a new observation is made. In cases where there are millions of landmarks to update the EKF's performance degrades substantially. FastSLAM comes as an improvement to this important aspect of computation cost. FastSLAM breaks up the problem of localisation and mapping into many separate conditionally independent problems. The joint SLAM state can be factored into a vehicle component and a conditional map component as described by the following eqn. 2.23

$$P(\mathbf{X}_{0:k}, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = P(\mathbf{m}|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k}P(\mathbf{X}_{0:k}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{2.23}$$

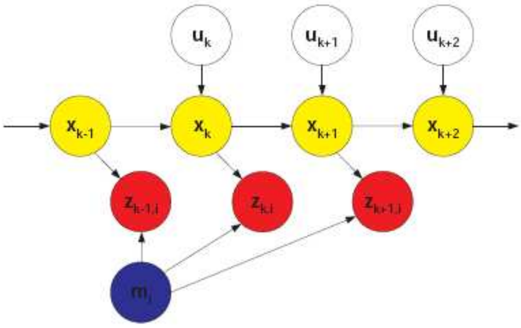


Figure 2.10: Graphical model of the FastSLAM algorithm. The past pose estimates are known, then the map states are independent since observations are conditionally independent. Figure from Durrant-Whyte and Bailey (2006).

The probability distribution is on the trajectory $\mathbf{X}_{0:k}$ rather than the single pose $\mathbf{x}_k$ because when conditioned on the trajectory the landmarks become independent (Durrant-Whyte and Bailey, 2006) as shown in Fig. 2.10. This is the key idea behind FastSLAM that the map is represented as a set of independent Gaussians with linear complexity.

The main structure of the FastSLAM is a Rao-Blackwellized state where the trajectory is represented by weighted samples. Thus, the set $\{\mathbf{w}_k^{(i)}, \mathbf{X}_{0:k}^{(i)}, P(\mathbf{m}|\mathbf{X}_{0:k}^{(i)}, \mathbf{Z}_{0:k})\}_i^N$ represents the joint distribution at time $k$ where the map accompanying each particle is composed of independent Gaussian distributions (Durrant-Whyte and Bailey, 2006), described by eqn. 2.24,

$$P(\mathbf{m}|\mathbf{X}_{0:k}^{(i)}, \mathbf{Z}_{0:k}) = \prod_{j}^{M} P(\mathbf{m}_j|\mathbf{X}_{0:k}^{(i)}, \mathbf{Z}_{0:k}) \tag{2.24}$$

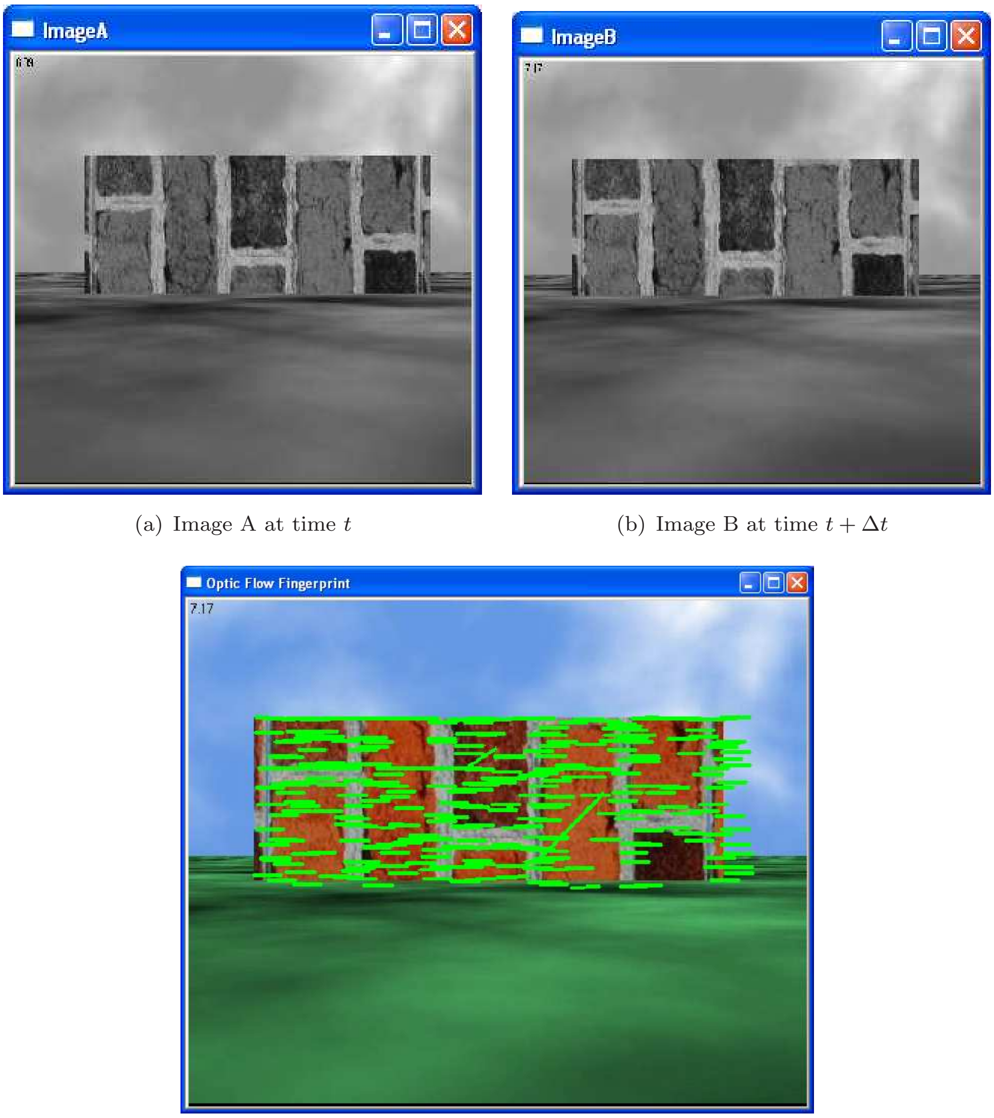
Recursive estimation is performed by particle filtering for the pose estimates and the EKF for the map states. Thus, each observed landmark is processed individually as an EKF measurement update from a known pose leaving unobserved landmarks unchanged. One of the disadvantages of the FastSLAM algorithm, however, is a possible decrease in accuracy as a result of ignoring correlations in landmark locations which in turn is caused by inaccuracies in the vehicle's position.

## 2.10 Optic Flow in Robotics

Lately a growing number of autonomous vehicles have been built using techniques inspired by insects and, in particular, optic flow. One of the first works that studied the relation of scene geometry and the motion of the observer was by Gibson (1974). A large amount of work, however, has been focussed on obstacle avoidance using optical flow (Camus et al., 1996; Warren and Fajen, 2004; Merrell et al., 2004). The technique, generally, works by splitting the image (for single camera systems) into left and right sides. If the summation of vectors of either side exceeds a given threshold then the vehicle is about to collide with an object. Similarly, this method has been used for centring autonomous robots in corridors or even a canyon (Hrabar et al., 2005) with the difference that the summation of vectors must be equal in both the left-hand side and the right-hand side of the image. Ohnishi and Imiya (2007) utilise optical flow for both obstacle avoidance and corridor navigation. Madjidi and Negahdaripour (2006) have tested the performance of optical flow in underwater colour images.

In a recent work implemented by Kendoul et al. (2009) optic flow is used for fully autonomous flight control of an aerial vehicle. The distance travelled in this Unmanned Aerial Vehicle (UAV) is calculated by integrating the optic flow over time. A similar work for controlling a small UAV in confined and cluttered environments has also been implemented by Zufferey et al. (2008). Barron et al. (1994) discuss the performance of optical flow techniques and their comparison and focussed on accuracy, reliability and density of the velocity measurements. Other works employ optic flow methods for depth perception (Simpson, 1993), motion segmentation (Blackburn and Nguyen, 1995), or estimation of ego-motion (Frenz and Lappe, 2005).

A similar technique to optical flow developed by Langer and Mann (2003) called *optical snow* arises in situations where camera motion occurs in highly cluttered 3D environments. Such cases involve a passive observer watching the fall of the snow, hence the name of the method *optical snow*. Optical snow has been inspired by research in animals that inhabit in highly dense and cluttered environments, such animals include the rabbit, the cat, and the bird. The properties of the optical snow are that yields dense motion parallax with many depth discontinuities occurring in almost all image points. This comes in contrast to the classical methods that compute optical flow and presuppose temporal persistence and spatial coherence. Langer and Mann (2003) present the properties of optical snow in the Fourier domain and investigate its computational problems of motion processing. Figure 2.11(c) shows an example of optic flow emanating from Fig. 2.11(a) and Fig. 2.11(b).



(a) Image A at time $t$

(b) Image B at time $t + \Delta t$

(c) Optic flow between image A and image B

Figure 2.11: Optic flow example

## 2.11 Mathematical Foundations of the Optical Flow

This section describes the mathematics that underlie the optic flow algorithms, and in particular, the Lucas-Kanade (LK) algorithm (Lucas and Kanade, 1981). In order for the optical flow algorithms to perform well, some suitable images need to be chosen. This suitability refers to images that have high texture and contain a multitude of corners. Such images have strong derivatives and, when two orthogonal derivatives are observed then this feature may be unique, and thus, good for tracking. Tracking a feature refers to the ability of finding a feature of interest from one frame to a subsequent one. Tracking the motion of an object can give the flow of the motion of the objects among different frames. In Lucas-Kanade algorithm corners are more suitable than edges for tracking as they contain more information. For the implementation of the LK algorithm the OpenCV (2008) library has been used.

Lucas-Kanade algorithm is also referred to as *sparse optical flow* in contrast to *dense optical flow*. Sparse optical flow refers to an algorithm that uses local information that is derived from some small window surrounding each of the points of interest. This is a drawback of the sparse approach as large motions can cause points to move outside the search window. Dense optical flow refers to to an algorithm that uses global information (Horn and Schunck, 1981). Dense optical flow algorithms generally are more robust to fast motions but their drawback is the heavy computational cost. To overcome the problem of local information in LK, the *pyramidal LK* algorithm was developed. The pyramidal LK tracks points from the highest level (coarse) of the pyramid down to the lowest (fine) level, thus allowing large motions to be tracked. Lucas-Kanade algorithm is applied on two contiguous frames to infer optical flow. The size of the window as well as the number of points to be considered for matching within the two frames are the requirements of Lucas-Kanade algorithm.

The most commonly used algorithm for corner finding is the edge detector by Harris and Stephens (1988). The second-order derivatives taken at all points in an image is called the *Hessian* image and it is described by the following eqn. 2.25,

$$H(p) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}_p \tag{2.25}$$

where I is the intensity of the pixel and $\partial x$ and $\partial y$ are the horizontal and vertical displacements of the pixel. The second-order derivatives are useful as can detect non-uniform gradients. For Harris corner detector the autocorrelation matrix of the second derivatives is used to find corners where two large eigenvalues occur. The eigenvalues of the autocorrelation matrix are also invariant to rotation. The autocorrelation matrix is described by eqn. 2.26,

$$M(x, y) = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \tag{2.26}$$

where $\alpha$, $\beta$, $\gamma$, and $\delta$ are expressed by eqns. 2.27, 2.28, 2.29, 2.30,

$$\alpha = \sum_{-K \leq i,j \leq K} w_{i,j} I_x^2(x+i, y+j) \tag{2.27}$$

$$\beta = \sum_{-K \leq i,j \leq K} w_{i,j} I_x(x+i, y+j) I_y(x+i, y+j) \tag{2.28}$$

$$\gamma = \sum_{-K \leq i,j \leq K} w_{i,j} I_x(x+i, y+j) I_y(x+i, y+j) \tag{2.29}$$

$$\delta = \sum_{-K \leq i,j \leq K} w_{i,j} I_y^2(x+i, y+j) \tag{2.30}$$

The $w_{i,j}$ is a weighting term that can be uniform that is often used to create a circular window or Gaussian weighting. Harris's corner detector is taking the determinant of $H(p)$, subtracting the trace of $H(p)$ with some weighting coefficient and then comparing the difference to a predetermined threshold. Shi and Tomasi (1994) have extended Harris corner detector by taking into account that the smaller of the two eigenvalues is greater than a minimum threshold. Sobel operator is used to compute the second derivatives and then the required eigenvalues. In order to find the required corners, subpixel location is used that requires the dot product between two gradients to be equal to 0 (Bradski and Kaehler, 2008).

The optic flow algorithm of Lucas-Kanade presupposes three main criteria to produce satisfactorily results. These are:

I. Brightness constancy. The brightness of a pixel does not change from frame to frame, that is $I(x, y, t) = I(x+u, y+v, t+1)$.

II. Temporal persistence or small movements. The motion of the object that is tracked moves smoothly from frame to frame, that is $I_x u + I_y v + I_t = 0$, where $v, u$ are the $x, y$ components of the velocity $\vec{u}$.

III. Spatial coherence. Neighbouring points of a pixel that belong to the same surface have typically similar motion, and project to nearby points on the image plane (Bradski and Kaehler, 2008).

The equation in the second criterion is an under constrained equation since involves two unknowns for any given pixel and cannot be used to solve the motion of a pixel in the two dimensions. For this reason the third criterion is used as an assumption to solve the full motion of a pixel in the two dimensions. The third criterion assumes that the neighbouring pixels of any given pixel move coherently as they belong to the same object and project to the same image plane as the given pixel projects. Thus, for solving the problem in case, the brightness values of the neighbouring pixel are taken and solve a system of linear equations. Hence, if a 5 × 5 pixel window is taken a system of 25 linear equations needs to be solved. However, if the window is too small the *aperture problem* may be encountered where only one dimension of the motion of a pixel can be detected and not the two-dimensional. On the other hand, if the window is too large then the spatial coherence criterion may not be met. Moreover, the larger the window the greater the computational cost that entails. A window of 5 × 5 has been considered as it provides a balance between computational cost and accuracy. Nevertheless, the system that needs to be solved following a window of 5 × 5 pixels is expressed by the following eqn. 2.31.

$$\underbrace{\begin{bmatrix} I_x(p1) & I_y(p1) \\ I_x(p2) & I_y(p2) \\ \vdots & \\ I_x(p25) & I_y(p25) \end{bmatrix}}_{A=25\times 2} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u}=2\times 1} = - \underbrace{\begin{bmatrix} I_t(p1) \\ I_t(p2) \\ \vdots \\ I_t(p25) \end{bmatrix}}_{b=25\times 1} \tag{2.31}$$

The goal on the above system of linear equations is to minimise $||A\vec{u} - b||^2$ where $A\vec{u} = b$ is solved by employing least-squares minimisation as in eqn. 2.32,

$$(A^T A)\vec{u} = A^T b \tag{2.32}$$

where $A^T A, \vec{u}$, and $A^T b$ are equal to, eqn. 2.33,

$$\underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}}_{A^T A} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u}} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b} \tag{2.33}$$

and the solution to the equation is, eqn. 2.34

$$\vec{u} = \begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b. \tag{2.34}$$

If $A^T A$ is invertible, i.e., no zero eigenvalues, it means it has full rank (2) and two large eigenvectors. This occurs in images where there is high texture in at least two directions. If the area that is tracked is an edge, then $A^T A$ becomes singular, eqn. 2.35,

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} -I_y \\ I_x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.35}$$

where $-I_y, I_x$ is an eigenvector with eigenvalue 0. If the area of interest is homogeneous then $A^T A \approx 0$, that implies 0 eigenvalues. The pyramidal approach of the (LK) algorithm solves the optical flow problem at the top layer by tracking over large spatial scales and then as it proceeds downwards to the lower layers the velocity criteria are refined until it arrives at the raw image pixels. Figure 2.12 shows the pyramidal structure of the LK algorithm.

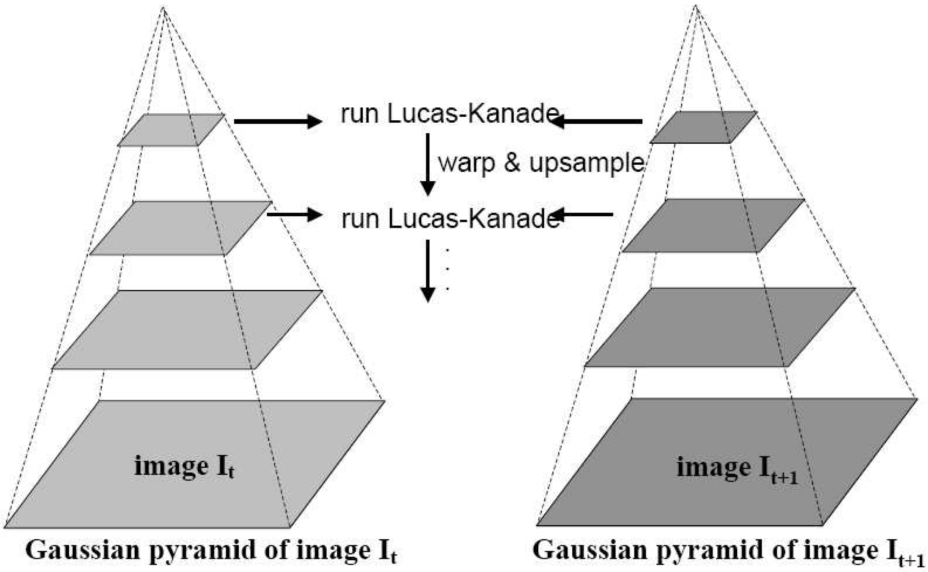


Figure 2.12: Pyramidal Lucas-Kanade structure. The LK algorithm begins at top coarse layer and the resulting motion estimates are used as a starting point for the lower ones. Figure from Bradski and Kaehler (2008).

## 2.12 Conclusion

This chapter discussed the methods used by animals for their navigation. It also discussed the corresponding implementations of these methods on robotic systems. It has been shown how homing is achieved by insects, and in particular ants and honeybees. As the field of experimental biology has significantly progressed the last few years, it has become feasible to understand better the mechanisms that underlie those animals. The methods that discussed in detail and which are related to this research were path integration, visual landmark recognition, optical flow, and searching. Although searching has not been implemented in this research, it has been discussed as future work can act upon it. In addition, a plethora of basic models and their variations were presented. In particular, the basic models of homing, that is ALV and snapshot were presented and their advantages as well as their limitations were addressed. Both the ALV and snapshot model require all landmarks to be present at the current and at the stored image. However, the two models can only be applied towards the end of the homing process. If they are applied in a large number of snapshots, then the calculated home vector can significantly deviate from the actual one. The main advantage of the ALV over the snapshot model, however, is that it does not require a matching and unwrapping of the images. All calculations are performed on the basis of vector components.

This chapter also discussed the TBL approach which has been used in this research to match landmarks that are near home position, thus, providing the robot with a supplementary information when close to its target. In addition, the Simultaneous Localisation and Mapping (SLAM) methodology as well as the optical flow technique were also discussed. Optical flow which has been a core part of this research has been presented with its advantages, limitations, and its applications. Thus far, optical flow has been used for obstacle avoidance and centring a robot in corridor-like paths. Having this in mind, this research has implemented a novel optical flow method for building topological maps of the environment based on the optical flow 'fingerprint' of the landmarks. The optical flow method meets all the objectives set in the previous chapter relating to biology and efficiency of the approach. The work that follows in Chapter four and Chapter five has drawn heavily on this chapter. The next chapter deals with the architectural framework of the simulated environment.

# Chapter 3

# The Simulation Environment

## 3.1 Introduction

This chapter describes the general simulation environment developed and used for the experiments. In particular, it discusses the framework of the Gazebo (2006) simulator and its supporting tools for the generation of the simulated environment. In addition, it presents the core characteristics of the sensors created and their functionality. Gazebo is part of the Gerkey et al. (2003) project and is developed in cooperation with Player and Stage projects whose development began in 2001.

## 3.2 Player and Stage Environments

Player (2006) is a socket-based device server that allows control of a wide variety of sensors and actuators (Gerkey et al., 2003). These could include bumpers, cameras, and laser range finders. Player executes on a machine that is physically connected to these devices. Client programs (i.e., programs written by users) can interact with Player via TCP socket interface and can control these devices (sensors, actuators) by exchanging messages through the TCP socket. Client programs can be written in any high-level language such as C/C++ or even Common LISP. The Stage is a two-dimensional simulator that interfaces with Player to simulate very large number of robots, in the order of hundreds. It is not accurate in terms of visual scenery since it provides only the two dimensions of it, but it provides a high fidelity representation of large teams of robots and their behaviour, e.g., kinematics. An example of the Stage simulator is depicted in Fig. 3.1. In addition to Player-stage, the *breve* simulation environment has been used for the generation of landmarks (Klein, 2002).

The following code depicts an example driver, called configuration file, for the player stage simulator. In the code a stage driver is created with a 2D position information as well as a laser range scanner. The world file which describes the simulation environment is loaded through the configuration file.

```
# Loads the Stage plugin simulation driver
driver
(
  name "stage"
  provides ["simulation:0" ]
  plugin "libstageplugin"

  # Loads the named file into the simulator.
  # Worldfile defines the simulated world,
  # i.e., the sensors and actuators to be
  # simulated and the background image to load
  worldfile "pss.world"
)

# Create a Stage driver and attach position2d and laser interfaces
# to the model "robot1"
driver
(
  name "stage"
  provides ["position2d:0" "laser:0" ]
  model "robot1"
)
```

Player-stage has provided the general framework, that is the templates, in order for the various sensors and models to be developed. The general framework can be seen from the source code of the configuration file. There, the models to be used are defined as well as their parameters. The algorithms for localising and mapping an environment have been developed and are part of this research. The purpose of a simulator is to simulate a real environment and provide output based on the input parameters and the algorithms employed. Thus, the algorithm for obstacle avoidance using a laser range finder as well as the properties of the laser sensor have been developed and defined by the author of this research. The graphics of the laser rays and the environment are properties defined by the creators of the simulator.

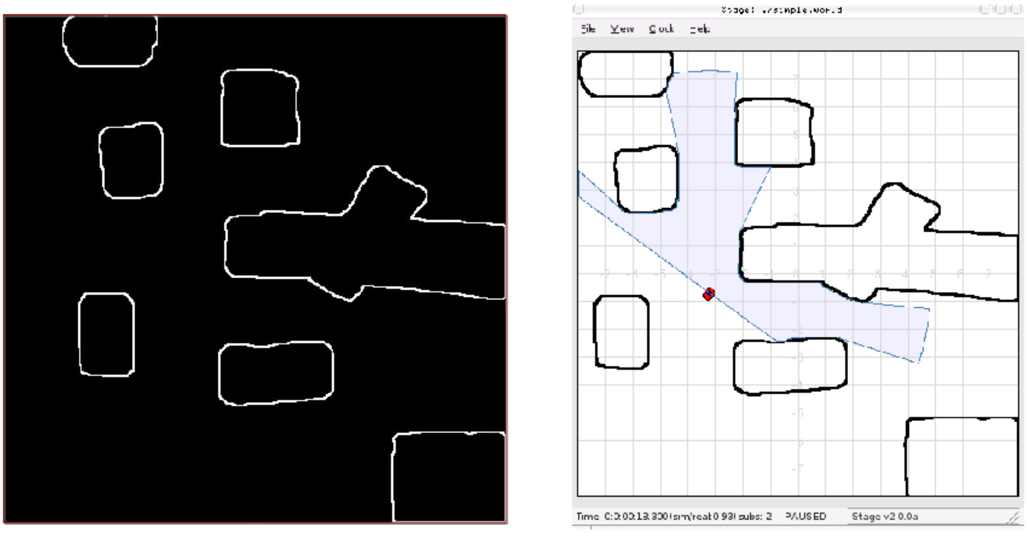


Figure 3.1: The Stage 2D simulator. The left-hand picture shows a simple 2D image and its representation in the Stage environment (right-hand side picture). Figure from Stage (2006).

## 3.3 Gazebo Simulation Environment

The Gazebo (2006) is a three-dimensional, open source simulator aiming at simulating robots mainly in an outdoor environment. Gazebo has been in use the last few years, and has rapidly evolved into a framework for simulating a population of robots in 3D structures. Whereas Stage is intended to simulate behaviour of very large populations of robots with low quality visual representation, Gazebo simulates the behaviour of small number of robots, in the order of tens, with high fidelity. The 3D representation of the environment and of the robots themselves with all the computational cost that this entails makes Gazebo ideal for simulating small teams of robots with high dynamics accuracy concerning robot-environment interaction.

All simulated objects in the environment have mass, velocity, friction, and a number of other attributes that allow them to behave realistically when a force is applied to them. The robot themselves are dynamic structures composed of rigid bodies connected via joints (Koenig and Howard, 2004). The following Fig. 3.2 shows the general framework of the Gazebo simulator and its components.

The *World* is a file written in XML in which all models, such as robot, objects, terrain, cameras, and other sensors are represented. In addition, some other parameters such as lighting, gravity, and resolution of images can be passed on to the Gazebo. Each model is composed of at least one body and any number of joints and sensors. The third party libraries, such as Open Dynamics Engine and GLUT, reside below the API in order to make models independent of any future changes of these third party libraries.

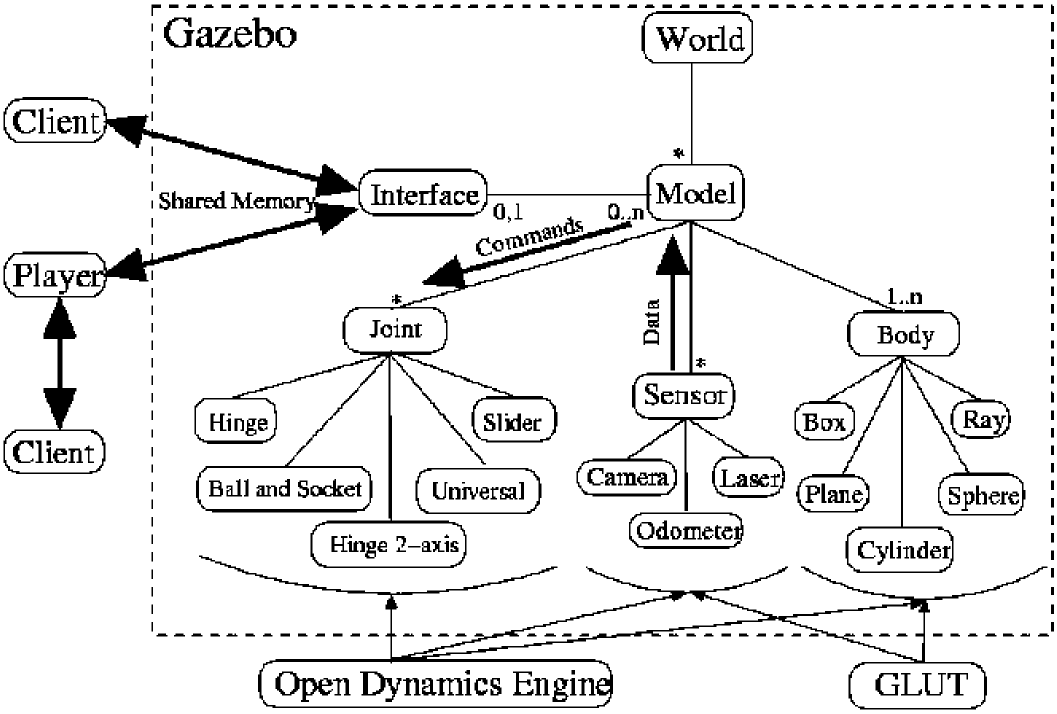


Figure 3.2: The general architectural framework of Gazebo simulator. Figure from Stage (2006).

The ODE (2006) (Open Dynamics Engine) is an open-source physics engine designed to simulate the dynamics and kinematics associated with the articulated rigid bodies. ODE provides features like joints, collision detection, mass and rotational functions. Gazebo utilises these features by providing a layer of abstraction situated between the ODE and Gazebo models (Koenig and Howard, 2004). OpenGL (2006) and GLUT (OpenGL Utility Toolkit) provide the visualisation tools for Gazebo. Scenes rendered using OpenGL are displayed in windows created by GLUT. The replacement of both the physics engine and the graphics engine can be done easily since they provide an abstraction to the Gazebo simulator, i.e., lying under the API interface. Gazebo can be used in conjunction with Player. Player and Gazebo can exchange messages and communicate in general through an interface. Gazebo can access all sensors provided by Player through shared memory. All three programs (Player, Stage, and Gazebo) have identical interfaces so programs written for one simulator can be employed unchanged in real-world robots.

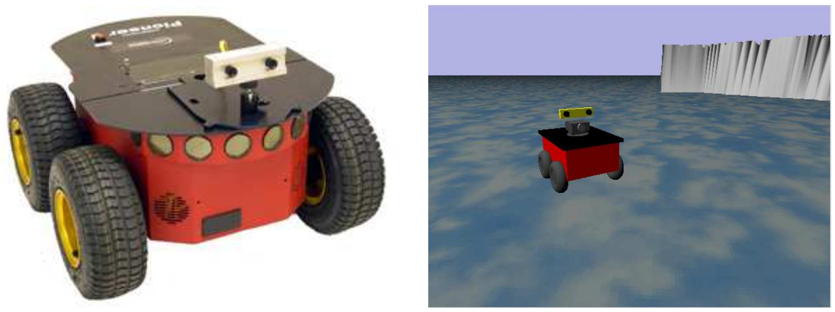
Following is an example code of the gazebo simulator which describes the simulated environment. In this code it can be seen the instantatiation of a geometrical object with its properties, that is, shape, size, colour, and position in space. Apart from the creation of the robot, the terrain with the ground floor is described in the following xml code.

```
# definition of the ground plane
# with the colour property and the
# loading of the respective texture file.
<model:GroundPlane>
  <id>ground1</id>
  <color>0.3 0.3 0.3</color>
  <textureFile>pattern.ppm</textureFile>
</model:GroundPlane>

# definition of the terrain.
<model:Terrain>
  <xyz>0.000 0.000 -0.100</xyz>
  <terrainFile>example.gzb</terrainFile>
</model:Terrain>

# definition of the the Pioneer2AT robot
# and its coordinates in space.
<model:Pioneer2AT>
  <id>robot1</id>
  <xyz>-0.000 0.000 0.001</xyz>

# definition of a geometric object with
# properties of shape, size, and colour.
<model:SimpleSolid>
  <xyz>2.500 0.000 0.000</xyz>
  <shape>box</shape>
  <size>2 2 2</size>
  <color>0.3 1.0 3</color>
</model:SimpleSolid>
```

In gazebo environment the various kinematics and collision properties have been defined by the owners of the simulator. As with Player-stage, the templates for sensors as depicted in the world file are part of the simulator. The algorithms by which the robot avoids obstacles, performs optical flow, localises itself and creates map are part of this research. The 3D landmarks used for this work are all stationary. Some of them have been textured in order to simulate a realistic object while others have been left with no texture but are in colour, in order to validate the biological models as they are described in Chapter 2, that is simple, colour geometrical objects.

### 3.3.1 Models and Sensors

The most basic model that was created for the purposes of this work is a *Pioneer 2AT* robot. Figure 3.3(a) shows the real robot and Fig. 3.3(b) depicts the modelled one. The model is a four-wheel robot equipped with a pan, tilt, and zoom stereo head, a Sony VID30 camera, and a sonar sensor. Various other parameters are controllable like lighting conditions, horizontal field-of-view, baseline and others. The majority of these parameters are handled through the *World* file.



(a) Pioneer 2AT robot. Figure from MobileRobots (2006)

(b) Modelled Pioneer 2AT robot with sensors

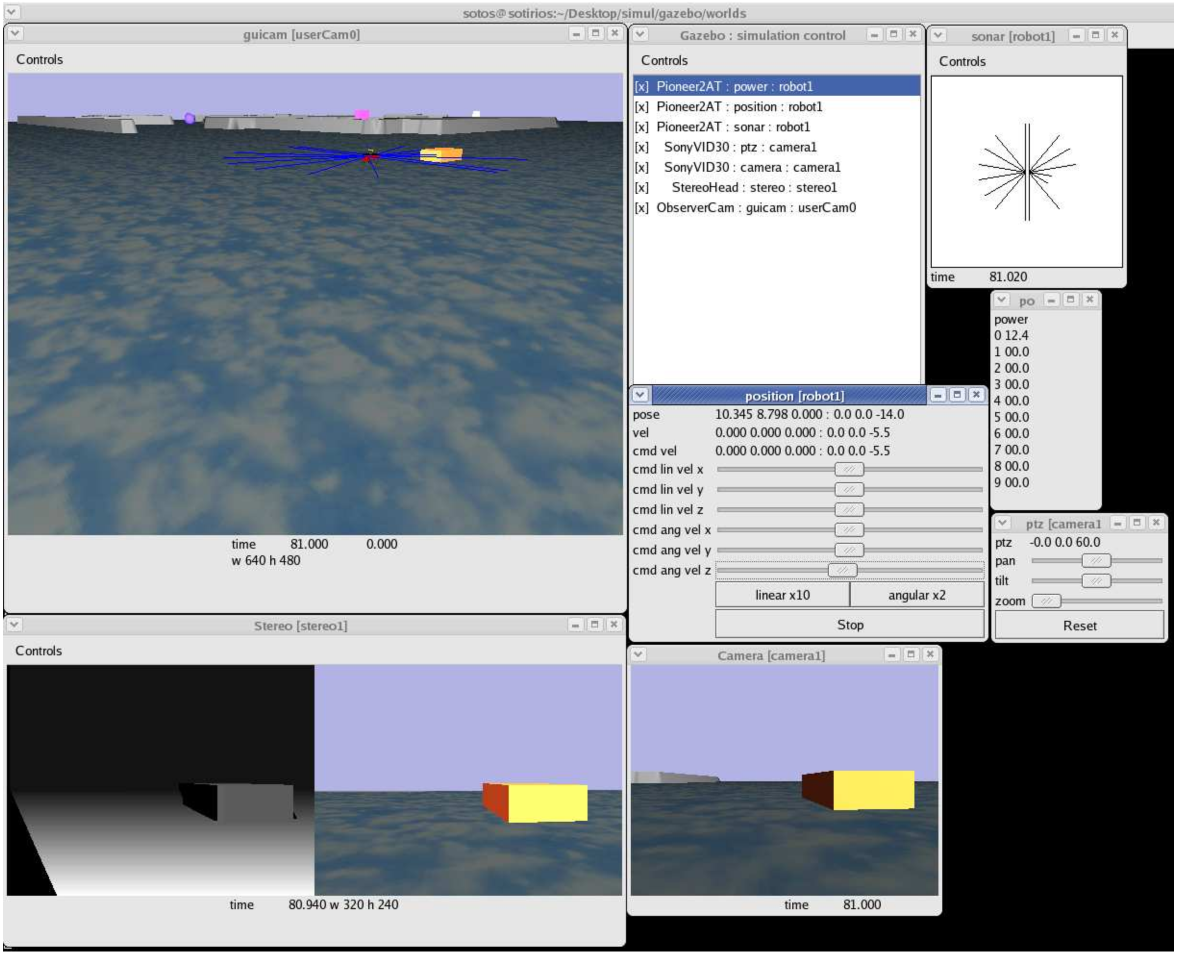
FIGURE 3.3: Pioneer 2AT robot

Figure 3.4 shows a snapshot of the Gazebo environment and the models employed. In the working environment, one can discern a maze and four distinct landmarks of different colour and shape. One can also see the rays emitted by the sonar sensor and its corresponding control interface.

The maze in the example of Fig. 3.4 was created with the *gzbuilder* utility (provided by Gazebo) that inputs a binary 2D structure like the one in Fig. 3.5(a) and outputs a 3D image loadable by the terrain model. The image must comply with the format rules of GDAL (2006) library. The 3D output structure as seen from the observer camera model is shown in Fig. 3.5(b). The maze created is in grey colour. The objects or landmarks can later be added to the environment.

## 3.4 Conclusion

This chapter discussed the general architectural framework of the simulator and the tools used for the creation of the 2D and 3D structures that has served as the platform for the development and testing of the navigation strategies. Player (2006) and Gazebo (2006) simulators have provided the ideal environment for the simulations of this work. In this chapter it was also shown how sensors and actuators are defined in a text file which are then loaded for simulation. Player-stage and gazebo simulators are open-source simulators that provide a plethora of properties for interacting a simulated device with



```
[x] Pioneer2AT : power : robot1
[x] Pioneer2AT : position : robot1
[x] Pioneer2AT : sonar : robot1
[x]   SonyVID30 : ptz : camera1
[x]   SonyVID30 : camera : camera1
[x]     StereoHead : stereo : stereo1
[x] ObserverCam : guicam : userCam0
```

```
time      81.000        0.000
w 640 h 480
```

```
time      81.020
```

```
power
0 12.4
1 00.0
2 00.0
3 00.0
4 00.0
5 00.0
6 00.0
7 00.0
8 00.0
9 00.0
```

```
pose          10.345 8.798 0.000 : 0.0 0.0 -14.0
vel           0.000 0.000 0.000 : 0.0 0.0 -5.5
cmd vel       0.000 0.000 0.000 : 0.0 0.0 -5.5
cmd lin vel x
cmd lin vel y
cmd lin vel z
cmd ang vel x
cmd ang vel y
cmd ang vel z
```

```
ptz    -0.0 0.0 60.0
pan
tilt
zoom
```

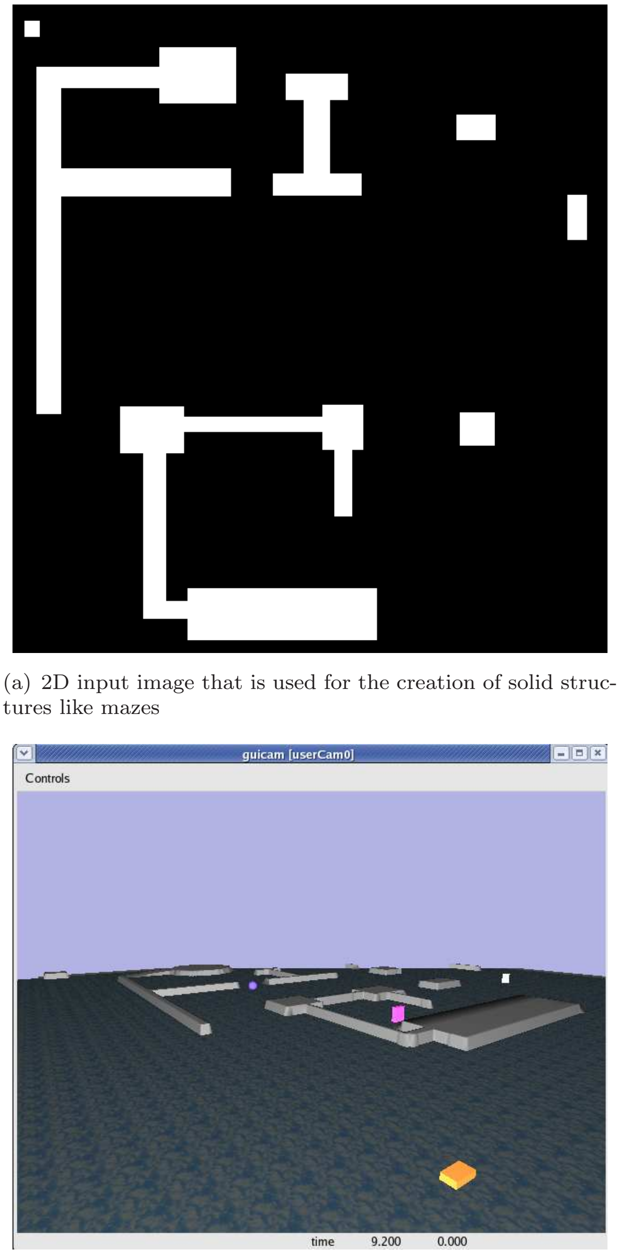
```
time      80.940 w 320 h 240
```

```
time      81.000
```

Figure 3.4: The working environment of Gazebo simulator depicting a Pioneer 2AT robot with a sonar sensor, a PTZ stereo head, and a Sony VID30 camera.

the environment. One of the main advantages of those simulators is that the algorithms developed can easily be ported to a robot with identical sensors. Because of their widespread use, a large number of simulated devices, from cameras to grippers, can be found on their online repository.

The following chapter discusses the algorithms developed for localising a robot within an environment while at the same time mapping that environment. The first part of Chapter 4 presents an algorithm for localisation and map building based on a laser range finder whose field of view is 240°. An inherent property of this process is obstacle avoidance. The second part of this chapter is dedicated to adding a camera sensor that can support the localisation and mapping process. Furthermore, a comparison is given between two different methods for calculating depth, that is the distance between an object and the camera, that is used to support localisation within specific environments, such as corridors. Chapter 4 and Chapter 5 comprise the research part of this work.



(a) 2D input image that is used for the creation of solid structures like mazes

(b) 3D output image with landmarks and terrain features added as seen from the observer's camera

FIGURE 3.5: 2D input image and 3D output image

# Chapter 4

# Metric Maps

## 4.1 Introduction

This chapter discusses the development and implementation of a parsimonious algorithm using a laser range scanner for localisation and mapping. The implementation of this algorithm produces metric maps which are essential for the navigation process, and in particular, homing. Metric maps are used in conjunction with topological maps which are discussed in the next chapter in order to infer distances between various landmarks. In graph theory terminology, metric maps would act as edges while landmarks inferred by topological maps would act as nodes.

The assumption that is taken in this method is that the robot will be able to detect 'distinct' landmarks in the environment. In cases this does not hold the robot can use its visual sensor to estimate depth between a landmark and the robot, thus supporting the localisation process. For estimating depths, two different strategies have been used, that is least squares and optical flow. A comparison between these approaches as well as a mixed approach is presented in this chapter.

## 4.2 Localisation and Mapping Using a Laser Range

In this work no prior information about the environment is assumed. The only hypothesis made is that the robot knows where it should head to. Thus, the robot is fed with a goal that has to be sought. However, even this hypothesis can be ruled out so long as the robot navigates without any restrictions, i.e., without having to pursue an effective path. In such a case the robot can navigate towards places where there are distinguishable points by the laser finder, for example corners, and drive among such 'places'.

In this chapter the navigation method presented is based on the idea of Vector Field Histogram (VFH) (Ulrich and Borenstein, 1998), a real-time motion planning and obstacle avoidance algorithm. The method that is presented in this chapter has adopted the idea of VFH in order to plan an obstacle-free path using a lase range finder. Nevertheless, the main significance of this research is the localisation of the robot and mapping of the environment using the same laser range finder. The robot scans the environment using the laser sensor and based on the measurements taken an efficient path is sought. In order to infer an efficient path the target point is assumed to be known. Therefore, a supposition is made that the goal position is known by its $x, y, \phi$ coordinates or that the goal position can be seen through a vision sensor and an estimate of its distance or direction can be taken. Next, the robot will try to identify 'distinctive' landmarks in the scanned environment; these are mainly corners, and can be identified by the range of the neighbouring ray values. The robot will thus extract the shortest obstacle-free path to the target defined by the rays of the laser scanner. While traversing the chosen route, the landmark is tracked down by the laser scanner's rays until it is 'met' by the side rays of it. The side rays of the laser scanner are used to identify the landmark as they form a 90° angle with the chosen path. The sides of the right triangle formed by the side rays of the scanner, the landmark, and the position of the robot at which the landmark was first detected are computed with the trigonometric functions.

The process of finding a 'distinctive landmark' is critical for the path to be extracted as is needed to be as close to the autonomous robot as possible. Figure 4.1 demonstrates this process where the robot at time $t_0$ scans the environment and selects the nearest landmark to it, depicted by the bold scan line. At time $t_1$ the robot is scanning again the environment using the laser scanner and an updated second path is chosen to reach the target at point denoted by the 'X' symbol. The robot could have missed reaching the target and ended up in a blind alley had it not selected a nearby landmark.

The primary purpose of this work is to tackle one of the core problems in robotics science, namely localisation and mapping using minimal sensing. It is not uncommon that a critical sensor to the system may fail and thus having to bypass it using alternative sensors and methods. Such cases become of high importance if the autonomous robot acts in remote or hostile environments. Moreover, sensors like GPS cannot operate in environments like, indoors, underwater, or the outer space. Therefore, the localisation and mapping problem is approached using only a laser scanner sensor to infer accurate maps of the environment while at the same time localising the robot. Later, in the chapter it is discussed how the addition of a vision sensor can make this algorithm more robust and reliable. It is also presented how the two different visual methods, a least squares and optical flow, can estimate depth.

### 4.2.1 Methodology

In this section are described the methods that have been followed to tackle the problem of localisation and mapping (Diamantas and Crowder, 2009). No *a-priori* knowledge

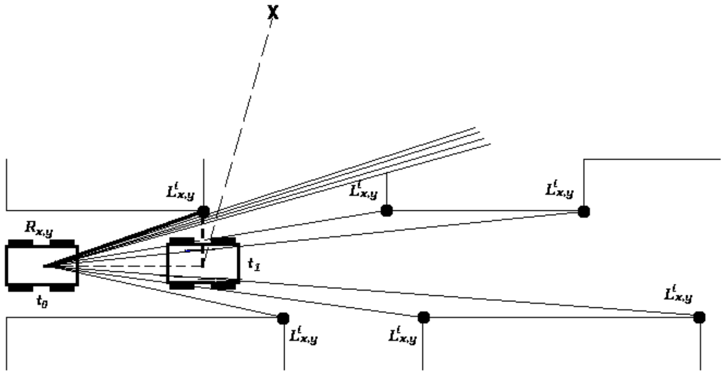


FIGURE 4.1: Robot having selected the nearest landmark (in bold line) which leads to an efficient path to the target point ‘X’.

of the environment is provided. However, as already stated a goal-seeking approach is adopted and a competent path is selected which is defined by the laser scanner rays. The map of the environment is initially empty and is built up as the navigation of the autonomous robot proceeds. Figure 4.2 depicts the structure of the localisation and mapping algorithm.

During the first phase, the robot collects raw data from the laser range finder, and in the next one the obstacles of the environment are expanded, Fig. 4.3, according to the radius, $r$, of the robot using the equation of the circle eqn. 4.1,

$$(x-h)^2 + (y-k)^2 = r^2, \tag{4.1}$$

where $h$, $k$ is the centre of the circle $C$, in this case the point at which the ray of the laser hits on the obstacle, and the equation of the line eqn. 4.2,

$$y = mx + b, \tag{4.2}$$

that represents a laser ray. The slope, (eqn. 4.3), of the line, i.e., ray is computed,

$$m = \frac{C_k - R_y}{C_h - R_x}, \tag{4.3}$$

by knowing the current position of the robot $R_{x,y}$ and the end position of the ray, i.e., $C_{h,k}$. Next, eqn. 4.2 is substituted into the second part of eqn. 4.1 to yield eqn. 4.4,

$$(x-h)^2 + (mx+b-k)^2 = r^2. \tag{4.4}$$

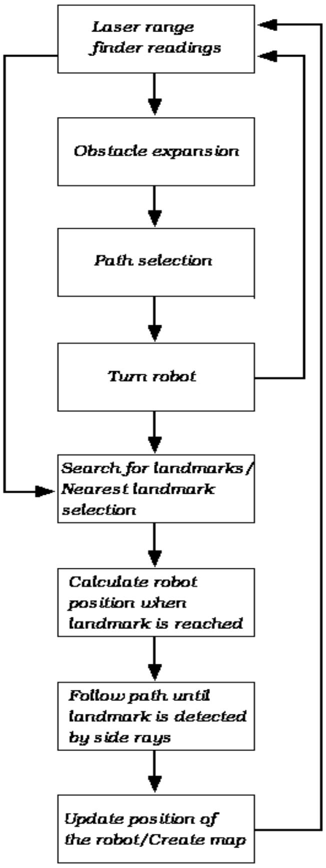

Laser range finder readings
Obstacle expansion
Path selection
Turn robot
Search for landmarks/ Nearest landmark selection
Calculate robot position when landmark is reached
Follow path until landmark is detected by side rays
Update position of the robot/Create map

FIGURE 4.2: Flowchart of the various stages of the localisation and map building algorithm.

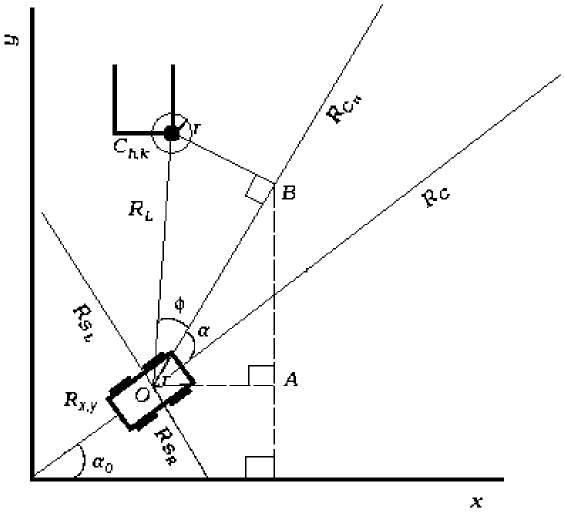


FIGURE 4.3: Robot having identified a new path, $R_{C_n}$, and a landmark, $C_{h,k}$. Localisation coordinates are computed from the triangles $OBC$ and $OAB$.

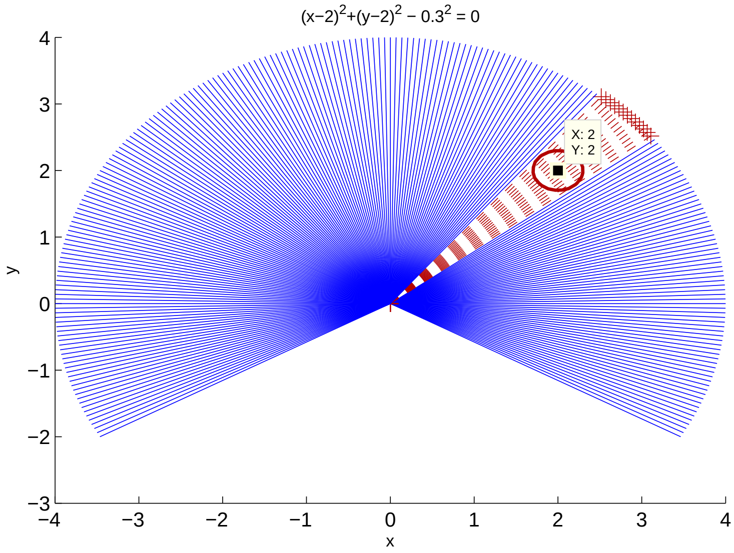


Figure 4.4: A simulated scan sample with a detectable area of 240° and a detectable distance of 4.0 metres. The point at $x = 2$, $y = 2$ is expanded by 0.3 metres which is equal to the radius, $r$, of the robot. The marked laser rays (shown in red dashed lines) are thus do not provide a safe path for navigation. If a path curved by the red lines is followed the robot will collide with the obstacle.

Thus, from eqn. 4.4 a quadratic equation is formed (eqn. 4.5),

$$ax^2 + bx + c = 0, \tag{4.5}$$

which is solved in order to obtain the points in which the rays intersect (or hit) the obstacles expanded by the radius $r$. Thus, if discriminant, $\Delta > 0$, then the ray intersects the expanded obstacle in two points, whereas if $\Delta = 0$, there exists a tangent ray to the circle. If $\Delta < 0$ then the ray is not intersecting any point of the circle formed by the expansion of the obstacle. Figure 4.4 shows the rays of the laser scanner one of which hits at an obstacle with coordinates $x = 2$, $y = 2$. The circle represents the expansion of the obstacle by the radius, $r$, of the robot, in this case 0.3 metres. The marked lines representing the laser rays that fall within the expanded obstacle do not thus provide a safe path for the robot. All other rays, hence paths, would provide a safe path if they were to be taken. Figure 4.5 shows the outcome of applying the above algorithm to the laser data.

Upon expansion of obstacles the path selection module takes over, Fig. 4.2, and an efficient path is chosen based on the proximity of the laser ray to the target point. As discussed earlier, the coordinates of the target are known to the system. This however, could have been omitted had it been used a compass or a vision sensor to know at any instant of time the direction to the target. So long as the path is selected, the robot rotates around its axis until its central ray, $R_C$, Fig. 4.3, points to the selected path, $R_{C_n}$. The next step involves updating the laser data by taking a new laser scan after the robot has rotated by angle $\alpha$. At this point the landmark search module is taking over which identifies corners, jumps, and discontinuities in the laser scan. Such points act as 'landmarks' and are identified when a ray being equal to the maximum range, say set $A$, and its neighbouring $n$ rays do not belong to the set $A$, i.e., they hit on an obstacle. Therefore, the mathematical formulation of this is, $ray_x \in A$, and $ray_{x+n} \notin A$ or $ray_{x-n} \notin A$ constitute a 'landmark'. Using this technique the robot can distinguish between an obstacle and end of range provided that not all $n$ rays with maximum range belong to an obstacle.

Furthermore, the landmark search module will eventually select a single landmark at each time step for navigation. The selection of this landmark is based upon its proximity to robot location. This is happening because a shorter path to the target may be discovered after the approximation of the robot to the landmark. Therefore, the use the Euclidean distance, $d$, between the landmark location, ($L_x^i$, $L_y^i$), and the robot, ($R_x$, $R_y$) is employed to find the nearest landmark. Figure 4.1 shows an example of landmark selection among a set of detected landmarks. The Euclidean distance is described by eqn. 4.6

$$d = \min i \sqrt{(R_x - L_x^i)^2 + (R_y - L_y^i)^2}. \tag{4.6}$$

The area of searching for landmarks is less than 90° from the central ray, $R_C$, Fig. 4.3, as the robot is using its side rays, namely, $R_{S_L}$ and $R_{S_R}$, to detect the landmark and build a map. At this point a local frame of reference is used to calculate distances to landmarks. Figure 4.3 shows how the method works. After having inferred a new path the robot rotates by angle $\alpha$, that is, from $R_C$ to $R_{C_n}$. At this point its $R_L$ ray detects and selects landmark $C_{h,k}$. The sides of the triangle $OBC$ named $CB$ and $OB$ are calculated using the trigonometric functions $\sin\phi$ and $\cos\phi$.

Having found the coordinates of local frame of reference, they are then added up in order to find the global frame of reference. Again, Fig. 4.3 depicts the variables involved. This is done by adding the angle by which the robot has rotated, that is $\alpha$ degrees, to initial pose $\alpha_0$, thus inferring the pose of the robot. The projection of side $OB$ to the *x-axis* and *y-axis* is then found. The sides of the triangle $OAB$ are calculated with the *cosine* and *sine* trigonometric functions. Their outcome will thus give us the $x, y$ coordinates of the point $B$ which are added to the initial $R_{x,y}$ coordinates of the robot.

In order for the robot to follow the path from point $O$ to point $B$ the landmark at point $C_{h,k}$ is tracked down by the neighbouring rays of the $R_L$ ray as the robot proceeds. At point $B$, as Fig. 4.3 shows, the side ray, $R_{S_L}$, will detect the landmark by the range of the $R_{S_L}$ ray which should approximately be equal to $CB$ found earlier. Due to noise, it is quite probable that the landmark be missed. For this reason a laser scanner whose field-of-view (FOV) is 240° has been simulated thus if the landmark is missed by the

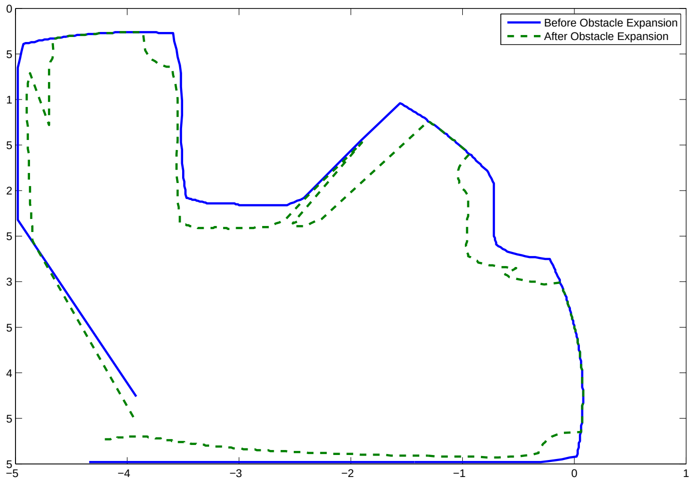


Figure 4.5: Expansion of obstacles. The obstacles are expanded by an amount equal to the radius, $r$, of the robot. This is essential in order for the robot to select an obstacle-free path.

side ray, $R_{S_L}$, it can be detected by the neighbouring rays which form an angle larger than 90°. Figure 4.6 shows this scenario.

At this time step the robot will localise itself with its global $x, y$ coordinates and create a new map of the environment. Having reached the final step of the algorithm, the robot initiates a new laser scan to proceed with the navigation process.

### 4.2.2 Simulation results

The localisation and mapping algorithm has been developed on the *Player/Stage* 2D simulation environment (Gerkey et al., 2003). The simulated devices consist of an *Erratic* mobile robot and a *HOKUYO URG04LX* laser range finder with 240° field of view, 4.0 metres maximum range, and 685 samples per scan. The angular resolution was set to 0.35°. The error in range conforms to the specification of the laser scanner and this is 1% of the detected distance. Figure 4.7 shows a snapshot of the simulated testing environment, the size of which is 10.0 metres by 10.0 metres. The robot is on the left lower corner with a delineation of the laser scanner.

Figure 4.8 shows the first scan at position $R_{x_0} = -3.92$, $R_{y_0} = -4.26$. The target position has been set at $x = 4.00$, $y = 4.00$. In this first laser scan there have been detected six landmarks of which, the nearest one, is selected for reference which is at position $x = -2.38$, $y = -2.06$. The robot will try to re-detect the landmark from position denoted by the symbol $\times$ at location $x = -2.09$, $y = -2.34$.

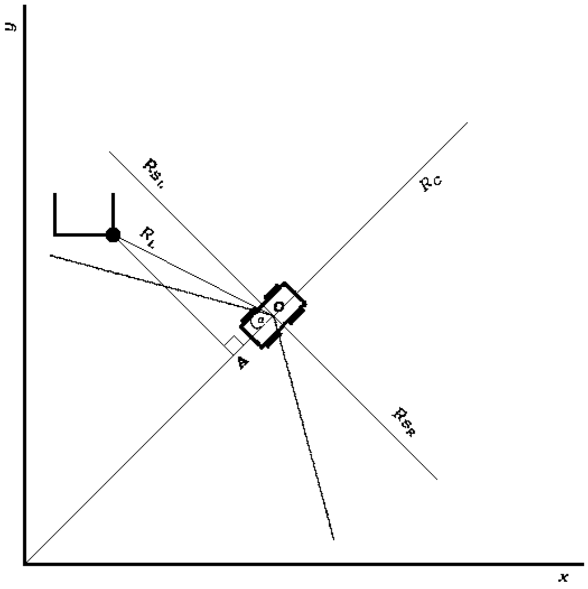


FIGURE 4.6: An example of how the location of the robot can be calculated after having missed the landmark by the side ray, $R_{S_L}$. The landmark is, however, still detectable by the neighbouring rays of the laser scanner.

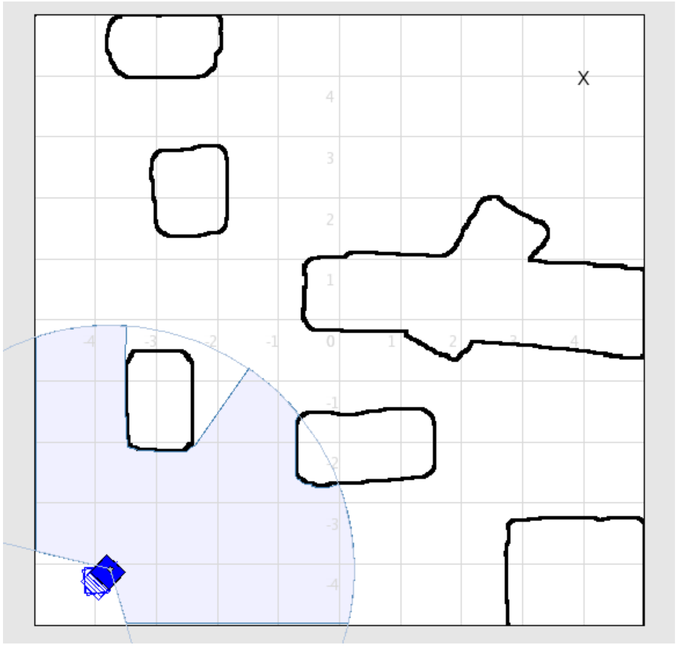


FIGURE 4.7: Snapshot of the simulated environment.

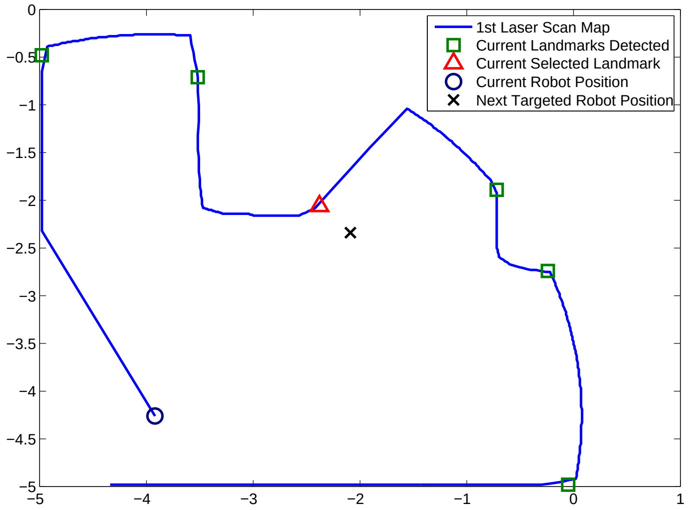


FIGURE 4.8: First scan of the environment. Six landmarks have been detected one of which is selected.

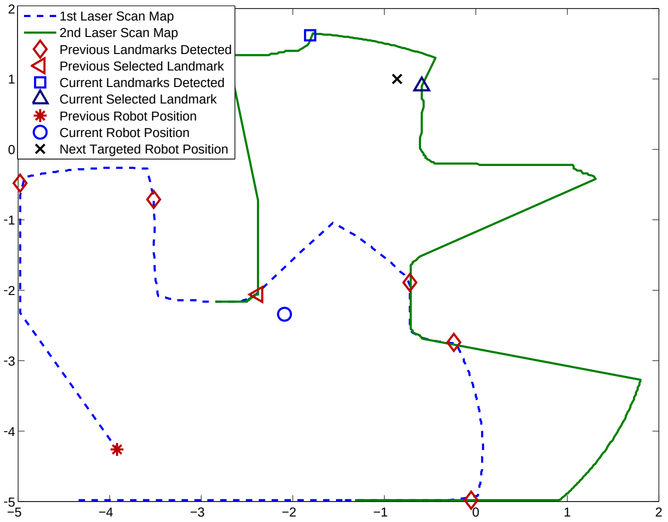


FIGURE 4.9: Second scan of the environment. Two landmarks have been detected one of which is selected.

In Fig. 4.9 the robot has successfully recognised the landmark and has updated its location coordinates; a new scan has been performed with two observed landmarks one of which again is selected for the navigation.

In Fig. 4.10 the robot has traversed the environment even more and has detected two other landmarks. At this point, although the target position is at $x = 4.00$, $y = 4.00$, the robot has selected its next targeted position, denoted by × symbol, to be further upwards

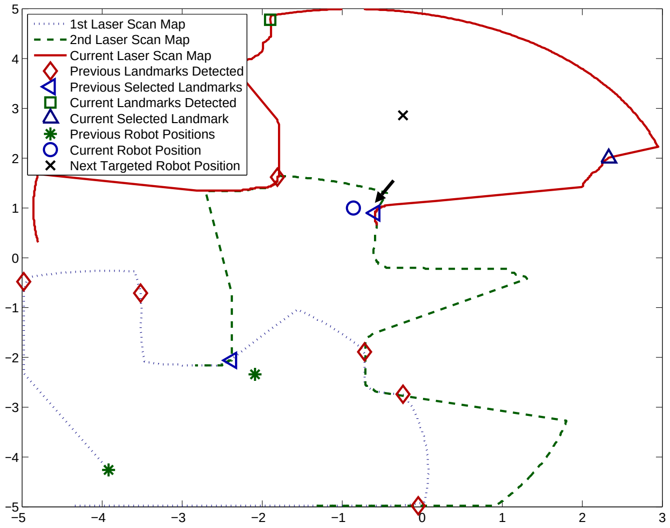


FIGURE 4.10: Third scan of the environment. Two landmarks have been detected one of which is selected. At this point the robot's laser finder is obstructed partly by the expansion of the obstacles at the right hand side. Thus, the next robot position is calculated to be further upwards instead of being slightly upwards.

instead of being on its right side. This occurs because the expansion of the obstacles obscures some laser rays that form a path in the right-hand side of the robot. The arrow shows the position, in particular the corner point, at which this occurs. Though thus far the selected path can be described as optimal, in cases such as this the optimality of the path is not guaranteed.

It should be noted that during the landmark selection process the map that does not contain the expanded obstacles is employed. The map with the expanded obstacles is used in the path selection process. That is why the landmarks selected are not artificially deformed. In Fig. 4.11 the robot has reached its desired position and a new scan has been taken which has detected two landmarks in the environment, one of which is again selected.

In Fig. 4.12 the robot has almost reached its target. The boldface × denotes the target position. However, in this figure it can be seen the influence of noise into the navigation process. There exists a map drift between the previous laser scan and the current one. This noise effect has caused the autonomous robot to select its next target point a little further away than the final target. The landmark selected in the previous time step, as shown in Fig. 4.11, is not a salient one; as it can be seen, the landmark at this point is more like a curved 'corner' and not a distinct corner.

Figure 4.13 shows all previous laser scans and the route of the robot as calculated by its localisation system and as it appears by the GPS system. From the graph it is seen that

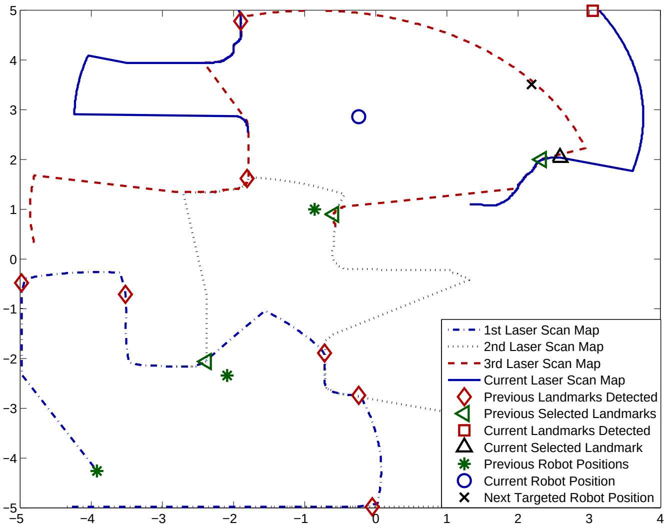


FIGURE 4.11: Fourth scan of the environment. Two landmarks have been detected one of which is selected.

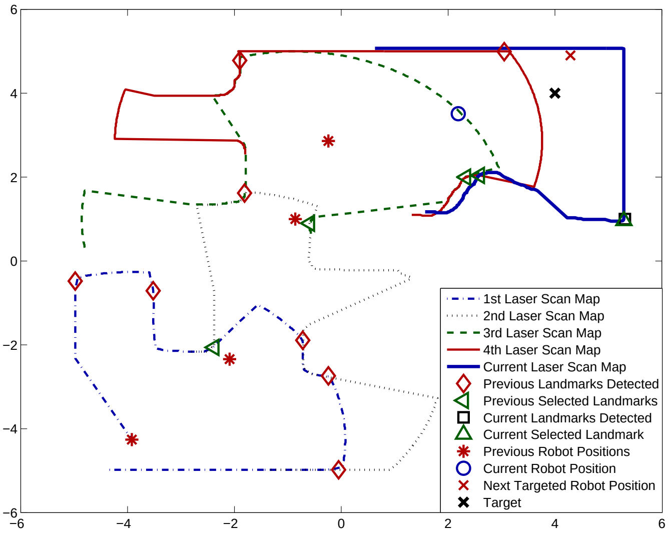


FIGURE 4.12: Fifth scan of the environment. One landmark has been detected and selected.

the autonomous robot has performed quite well in the first scans. Table 4.1 summarises the coordinates of the robot, the GPS, the selected landmark's as well as the coordinates of the next position to be reached. Nevertheless, there is a small drifting in the accuracy of the localisation system in the last phase of the navigation process caused mainly by the noise. In this last phase the landmarks selected, as can be seen from Fig. 4.13, are not prominent corners, but they rather occur in a curvy slope.

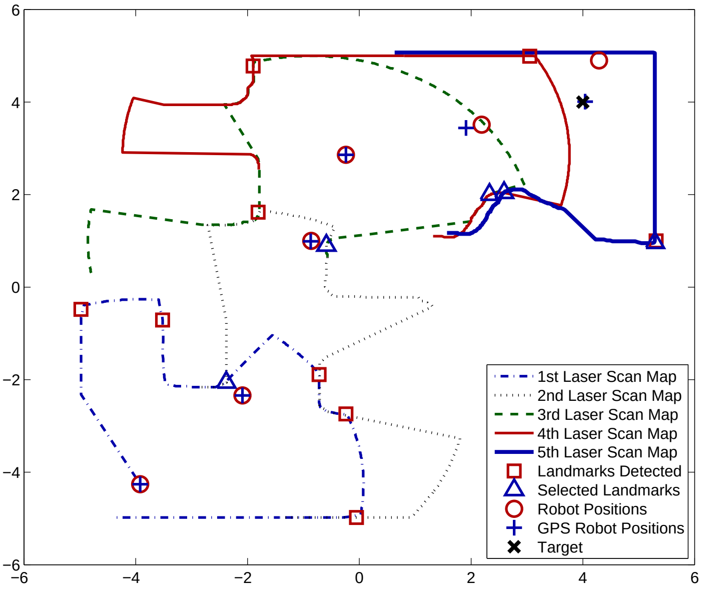


FIGURE 4.13: A performance comparison between the localisation system of the robot and the GPS.

TABLE 4.1: Localisation and Mapping Coordinates

| | Robot | GPS | Landmark | Location |
|---|---|---|---|---|
| 1st scan | (-3.92, -4.26) | (-3.92, -4.26) | (-2.38, -2.06) | (-2.09, -2.34) |
| 2nd scan | (-2.09, -2.34) | (-2.09, -2.34) | (-0.59, 0.9) | (-0.86, 1.0) |
| 3rd scan | (-0.86, 1.0) | (-0.87, 0.99) | (2.33, 2.0) | (-0.24, 2.86) |
| 4th scan | (-0.24, 2.86) | (-0.24, 2.86) | (2.59, 2.04) | (2.19, 3.51) |
| 5th scan | (2.19, 3.51) | (1.91, 3.44) | (5.29, 0.96) | (4.29, 4.9) |
| Location | (4.29, 4.9) | (4.04, 4.01) | | |

## 4.3 Navigation With a Vision Sensor and a Laser Range Scanner

The work in this section intends to expand the localisation and mapping algorithm by adding one more sensor, namely a camera. As described earlier, the localisation and mapping algorithm requires the extraction of distinct landmarks in order to accomplish its objectives smoothly. However, there are cases where there are no distinct landmarks and this is the case when a robot is passing through a corridor. If the algorithm is used outdoors then there may be cases where there are no corners nearby. For these reasons, the algorithm has been supplied with one more modality, the visual. For the solution of this problem a least-squares approach has been employed, that is, a visual landmark is tracked in the environment and the position of the landmark is inferred based on the orientation of the robot (Diamantas et al., 2010, In Press,a).

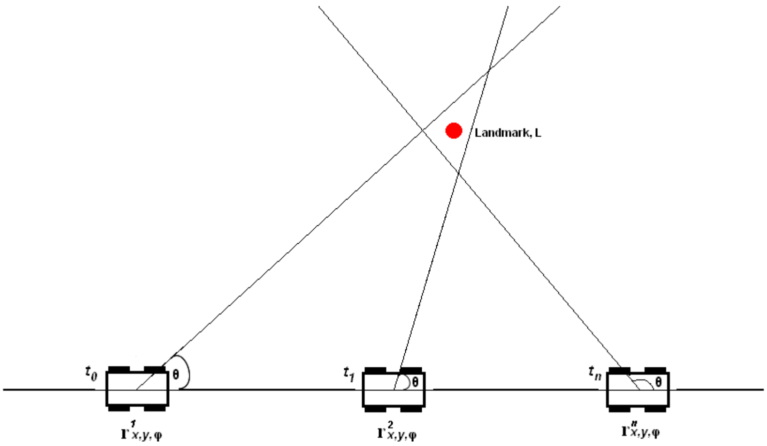


FIGURE 4.14: Robot snapshots of the landmark $L$ taken at different time steps $t$ and at different robot positions $r$.

### 4.3.1 Least Squares

A least squares approach is highly suitable for such type of cases as it can provide a relatively high estimate of the distance between the robot and a landmark with low computational cost. It is assumed that when the robot enters into a corridor its orientation does not change significantly, or it changes slightly. The input to the least squares is the $x, y$ coordinates provided by the laser finder as well as the angle $\theta$ between the robot and the landmark that is provided by the visual sensor. A similar method has been used by Boley et al. (1995) to infer the starting position of the robot. The odometry in their algorithm is provided by the dead reckoning. In our approach, however, the least squares algorithm builds upon the localisation and mapping model (Diamantas and Crowder, 2009). In other words, odometry is provided by means of the laser range scanner approach.

For the estimation of depth using the least squares approach a visual landmark is tracked in the environment and the position of the landmark is inferred based on the orientation and position of the robot. In the simulation, various snapshots of a landmark have been taken at equally spaced time steps and measure the angle $\theta$ between the robot position and the visual landmark, Fig. 4.14. Every snapshot represents a linear equation and the solution of the linear system gives the position of the landmark $L$. Thus, a set $\Omega_i$ is formed for every snapshot taken. Equations 4.7 and 4.8 show an example of two snapshots,

$$L \in \Omega_1 = \{h \in \Re^2 | \underbrace{(v_1 - r_1)^T}_{\alpha_1} h = \underbrace{v_1^T \cdot r_1 - ||r_1||^2 \}}_{\beta_1} \tag{4.7}$$

$$L \in \Omega_2 = \{h \in \Re^2 | \underbrace{(v_2 - r_2)^T}_{\alpha_2} h = \underbrace{v_2^T \cdot r_2 - ||r_2||^2 \}}_{\beta_2} \tag{4.8}$$

where $r_1$ and $r_2$ are the positions of the robot, and $v_1$ and $v_2$ are any two points on a line which is perpendicular to the line formed between the robot and the landmark, that is angle $\theta$. In essence, what needs to be found is the $L \in \Omega_1 \cap \Omega_2$. Figure 4.14 shows a robot at three different positions, $r^1$, $r^2$, and $r^n$ at corresponding time intervals $t_0$, $t_1$, and $t_n$, that also represent three different snapshots. The following equations, eqn. 4.9 and eqn. 4.10, show the process for a $n$ number of snapshots. Noise in the system is represented by variable $\epsilon_i$.

$$\begin{aligned} h\alpha_1 + \epsilon_1 &= \beta_1 \\ h\alpha_2 + \epsilon_2 &= \beta_2 \\ &\vdots \\ h\alpha_n + \epsilon_n &= \beta_n \end{aligned} \tag{4.9}$$

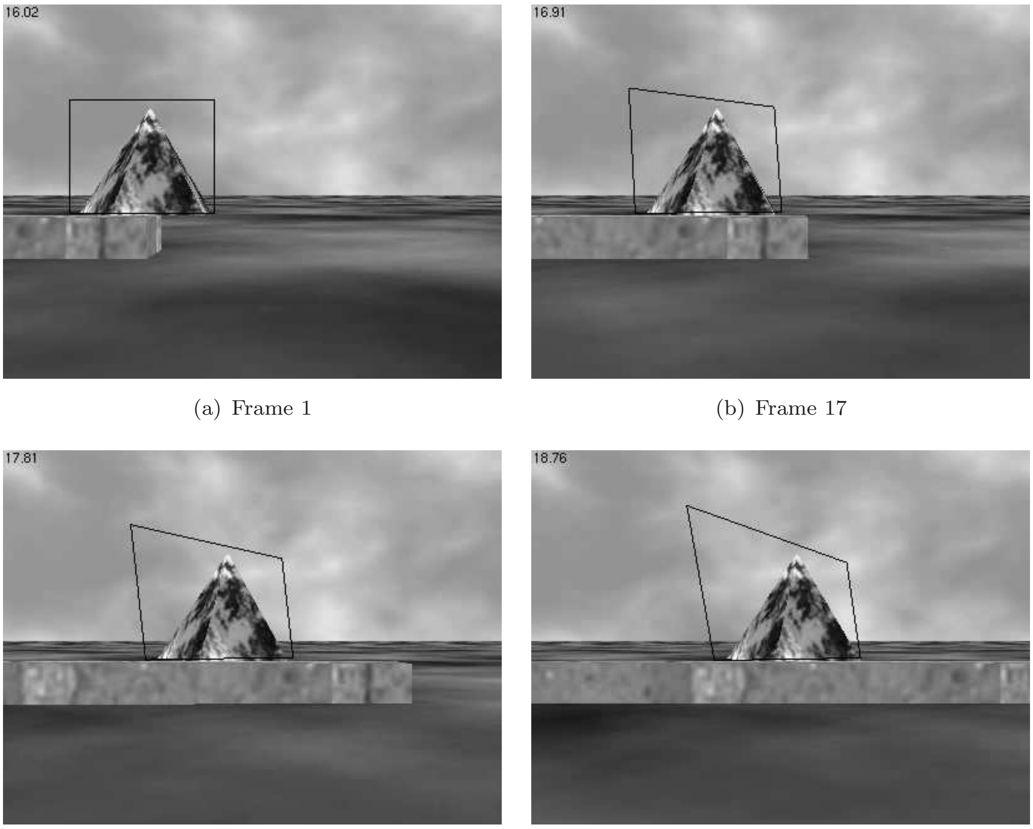
$$h \in argmin \sum_{i=1}^{n} (h\alpha_i - \beta_i + \epsilon_i)^2 \tag{4.10}$$

$$\underbrace{\left( \sum_{i=1}^{n} \alpha_i \alpha_i^T + \epsilon_i \right)}_{C} h = \underbrace{\left( \sum_{i=1}^{n} \alpha_i \beta_i \right)}_{\gamma} \tag{4.11}$$

$$h = C^{-1}\gamma \tag{4.12}$$

In eqn. 4.11 $C$ is a $2 \times 2$ matrix and $\gamma \in \Re^2$. The position of the landmark is, thus, given by $h$ where its y-axis element is considered, that is depth. In this experiment $n = 1000$ observations were used (drawn by a Gaussian distribution with $\mu = 11$, $\sigma = 3$) at varying distances between the landmark and the robot. They were performed 10 trials for each observation in order to estimate the mean position of the landmark. In addition, the error, $\epsilon_i$, in $\theta$ is uniformly distributed and it has been tested with three different deviations $\pm 1^\circ$, $\pm 3^\circ$, and $\pm 5^\circ$. Figures 4.15(a)-4.15(d) show the tracking of a landmark in 3D environment using the tracking algorithm of Benhimane and Malis (2007).

The graphs, Figs. 4.16(a)-4.16(d), depict the convergence of the least squares approach as the number of snapshots increase. Deviation represents the difference between the actual and the estimated distance of the landmark. In each of the four graphs three different error levels have been used. Furthermore, as it can be seen that a sensible estimate can be inferred with a small number of steps, that is snapshots. This has the advantage of low cost as a small number of linear equations needs to be solved. When the landmark is relatively close to the robot the depth estimation remains quite high and it drops as the distance between the robot and the landmark increases. Therefore,



(a) Frame 1

(b) Frame 17

(c) Frame 34

(d) Frame 51

Figure 4.15: Tracking of a landmark at different frames

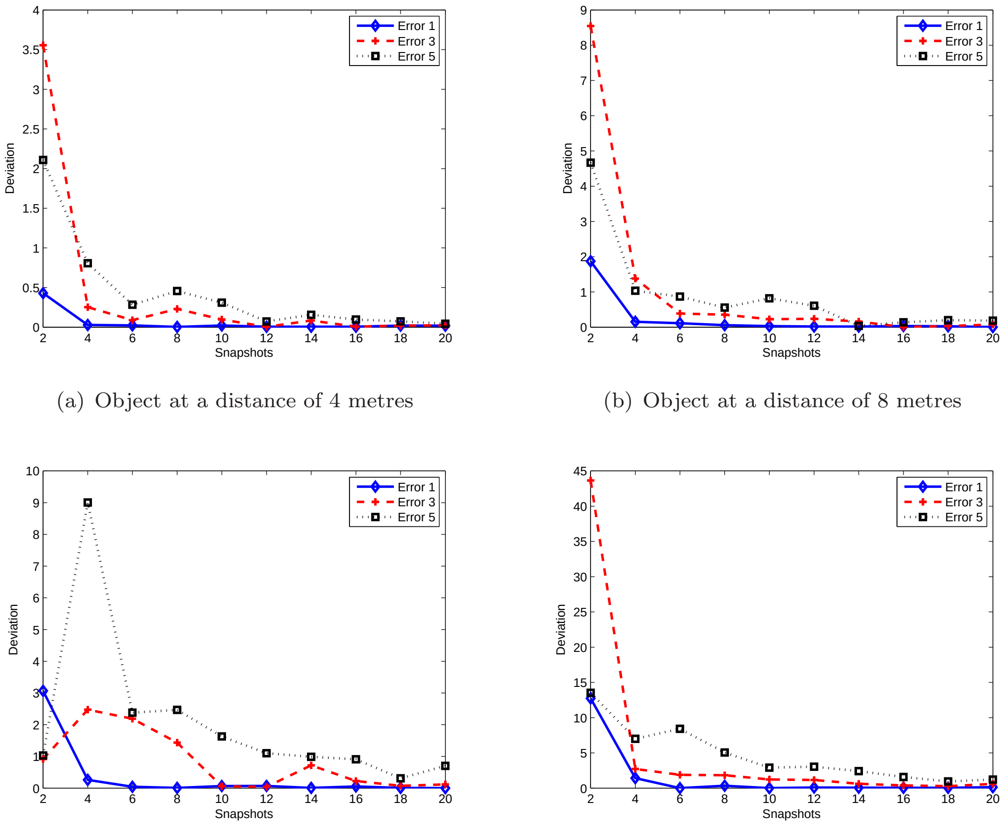
by knowing the position of the landmark, the robot is able to compute how much path it has traversed along its $x - axis$ using the trigonometric functions.

### 4.3.2 Optical Flow

In order for the robot to localise in an environment using optical flow vectors, a training data set of $n = 1000$ observations has been implemented where a vector is observed at varying distances between the robot and the landmark, and at varying velocities. Moreover, the distances and velocities chosen to create the training set approximate the real distributions of distance and velocity when a robot navigates in an environment. Thus, a joint probability distribution has been created by two continuous and independent variables, that is distance, $D$, and velocity, $E$, and is expressed by eqn. 5.4,

$$f_{D,E}(d, e) = f_D(d) \cdot f_E(e) \qquad \forall d, e. \tag{4.13}$$

The velocity (in $km/h$) and the distance (in $metres$) variables have been drawn from two Gaussian distributions with $\mu = 4$, $\sigma = 1$ and $\mu = 11$, $\sigma = 3$, respectively. The $n$ observations model the position of the optic flow vector in the plane under $n$ varying



(a) Object at a distance of 4 metres

(b) Object at a distance of 8 metres

(c) Object at a distance of 12 metres

(d) Object at a distance of 16 metres

FIGURE 4.16: Deviation (*metres*) between actual depth and estimated using least squares approach under different object distances.

distances and velocities. Figure 4.17 provides a pictorial representation of Gaussian distributions for velocity and distance.

One assumption that needs to be met in our method is that the majority of the vectors comprising a given landmark should have the same, or almost the same magnitude. In addition, the orientation of the camera on the robotic platform is perpendicular to the direction of motion so as a translational optical flow information is generated. During the testing of the optical flow algorithm the velocity of the robot is not known. The only information employed for inferring depth is the optical flow vectors of the landmark. The following section presents the results from the two strategies.

## 4.4 Comparing and Combining Optical Flow and Least Squares

In the least squares strategy, as it would be expected, the fit of the estimated distance to the actual distance increases as the number of snapshots increase and as the mean

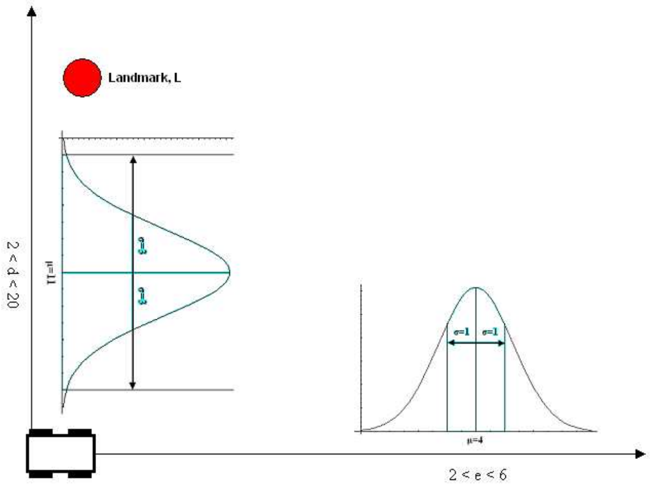


FIGURE 4.17: Gaussian distributions used for modelling distance, $D$, and velocity, $E$, in the optical flow strategy. Their mean and standard deviation are $\mu = 11$, $\sigma = 3$ and $\mu = 4$, $\sigma = 1$, respectively. The range of their values is $2 < d < 20$ ($metres$) for distance and $2 < e < 6$ ($km/h$) for velocity.

simulation error decreases. However, the larger the number of snapshots the greater the computational cost. Table 4.2 presents the $R^2$ statistic for various combinations of number of snapshots and mean simulation error. This $R^2$ statistic for $n$ snapshots and different error corresponds to the fit of the model $D = 0 + 1 \cdot E(D)$ where $D$ is the actual distance and $E(D)$ is the expected distance estimated through the least squares method. The minimum number of snapshots is 4 as this is the number required for the system to become overdetermined. Assuming that estimated distance is an unbiased estimate of the actual distance, if linear regression is used to estimate (for each combination of number of snapshots and mean simulation error) $a$ and $b$ in the function $D = a + b \cdot E(D)$, using the experiment's data, it should turn out that the best estimates are (approximately) $a = 0$ and $b = 1$. However, this is not the case as there seems to be a tendency for this method to systematically underestimate distance, especially when the mean simulation error is high and the number of snapshots low. The level of

TABLE 4.2: Performance of Least Squares method - $R^2$

| Snapshots $n$ | $R^2$ ($\epsilon_i \pm 1°$) | $R^2$ ($\epsilon_i \pm 3°$) | $R^2$ ($\epsilon_i \pm 5°$) |
|---|---|---|---|
| 4 | 0.971 | 0.137 | 0.000 |
| 6 | 0.991 | 0.722 | 0.000 |
| 8 | 0.996 | 0.883 | 0.336 |
| 10 | 0.998 | 0.940 | 0.649 |
| 12 | 0.999 | 0.967 | 0.801 |
| 14 | 0.999 | 0.979 | 0.881 |
| 16 | 1.000 | 0.987 | 0.915 |
| 18 | 1.000 | 0.991 | 0.944 |
| 20 | 1.000 | 0.993 | 0.959 |

underestimation is more obvious if in the previous function the constant is constrained to be 0. Interestingly, even though the bias decreases as the number of snapshots increases or the mean simulation error decreases, it is always significant as the standard error of the coefficient decreases too. In the following Table 4.3 the estimated coefficient of expected distance is presented for each combination of number of snapshots and mean simulation error. The confidence interval is 95%.

TABLE 4.3: Performance of Least Squares method - Coefficient

| Snapshots $n$ | Coeff. ($\epsilon_i \pm 1°$) | Coeff. ($\epsilon_i \pm 3°$) | Coeff. ($\epsilon_i \pm 5°$) |
|---|---|---|---|
| 4 | 1.017 | 1.166 | 1.726 |
| 6 | 1.013 | 1.130 | 1.386 |
| 8 | 1.009 | 1.084 | 1.239 |
| 10 | 1.006 | 1.057 | 1.162 |
| 12 | 1.004 | 1.042 | 1.118 |
| 14 | 1.004 | 1.033 | 1.089 |
| 16 | 1.003 | 1.026 | 1.073 |
| 18 | 1.002 | 1.021 | 1.059 |
| 20 | 1.002 | 1.018 | 1.049 |

In the optical flow strategy, regression analysis has been employed to estimate the distance between a landmark and the robot based only on the observed length of the optical flow vectors. Modelling the natural logarithm of distance as a linear function of the observed vectors' length is the optimal method to estimate distance. The regression formula that expresses this distance is described by eqn. 4.14,

$$\ln D_i = a + b \cdot len_i + \epsilon_i \tag{4.14}$$

where $\ln D_i$ is the expected distance, $a$ is the constant, $b$ is the coefficient, $len_i$ is the length of the optical flow vectors, and $\epsilon_i$ is the error.

Thus, linear regression is utilised to estimate the combination of $a$ and $b$ that minimises the squared error in eqn. 4.14. Equation 4.14 has also been employed to compute $R^2$ at different velocity intervals in order to adapt to realistic implementations. The outputs of these regression models are presented in Table 4.4.

TABLE 4.4: Performance of Optical Flow method

| Velocity $e$ $(km/h)$ | Constant $a$ | Coefficient $b$ | $R^2$ |
|---|---|---|---|
| $2 < e < 6$ | 2.96 | -0.075 | 0.6029 |
| $2 < e < 5$ | 3.03 | -0.09 | 0.6483 |
| $2 < e < 4$ | 3.08 | -0.11 | 0.7569 |
| $2 < e < 3$ | 3.1 | -0.143 | 0.8052 |

Obviously, velocity influences the length of the optic flow vectors and the smaller the range used the better the estimation of the expected distance. However, we avoided using velocity as a variable in the tests as this work is intended to be used in self-localisation systems where the position of the robot is provided only by means of a laser range finder. Hence, velocity cannot be estimated with a laser finder.

The results show that optical flow can perform better in cases where the mean error is high and a small number of snapshots have been taken in the least squares strategy. Nevertheless, when velocity range is large enough, optical flow does not perform satisfactorily enough when compared to least squares.

As these two strategies, namely optical flow and least squares, offer two independent estimations of the distance between the robot and the landmark, combining them should offer a more accurate estimate. Thus, an ordinary least squares (OLS) regression is employed to estimate a function that gives the most accurate measurement of the actual distance. Obviously, as the number of snapshots increases, the estimation becomes very accurate, but cost inefficient, too. Therefore, measuring distance based on the length of the vectors of the observed landmark offers little additional information, an estimation derived after a large number of snapshots was taken. However, it can significantly improve estimations that arise from a lower number of snapshots and therefore, provide higher accuracy, without sacrifising cost efficiency.

By way of example, the natural logarithm of distance has been modelled as a function of the estimated distance derived by four snapshots (sn4), mean error equal to three (e3), and the length of the observed vector. After experimenting with various functional forms of the two variables, various equations have been inferred which are described in Table 4.5. The dependent variable is *ln distance* and $R^2 = 0.8366$.

Table 4.5: Performance of Least Squares and Optical Flow methods

| Variable | Coefficient | T-Stat | P-Value |
|---|---|---|---|
| $ln(e3sn4)^2$ | 0.642 | 23.020 | 0.000 |
| $ln(e3sn4)^3$ | -0.141 | -18.040 | 0.000 |
| $ln(len)$ | 0.821 | 2.300 | 0.022 |
| $(ln(len))^2$ | -0.533 | -3.010 | 0.003 |
| $(ln(len))^3$ | 0.086 | 3.010 | 0.003 |
| constant | 0.587 | 2.410 | 0.016 |

Obviously, increasing the number of snapshots decreases the additional value of the optical flow method. Thus, if a high number of snapshots is taken there is no need to consider information derived by optical flow. As a consequence, whether to estimate the distance based on optical flow information, through observed snapshots or via a mixed approach, is a trade-off between desired accuracy and cost efficiency. Similarly, as the number of snapshots increase, the optic-flow related variables' significance statistics will be reducing as the additional value of this information becomes insignificant, if a high number of snapshots is taken.

## 4.5 Conclusion

In this chapter the problem of localisation and mapping has been addressed. The key idea behind our work is that no prior knowledge of the environment is used nor any odometry information. In fact, the odometry information is provided by a laser range finder. Its main purpose is to localise a robot and map the environment while at the same time selecting efficient paths for driving the robot to a target place. An inherent behaviour is obstacle avoidance. This is performed while path planning obstacle-free routes to desired locations. The requirements of the method are kept at minimum since only one sensor is employed. Nevertheless, the limitation of the approach is that it is dependent on the structure of the environment. An environment that contains distinct geometrical objects, can produce better results than an environment that contains curved surfaces. In cases where there are no detectable features an extended approach of the afore-mentioned method has been employed that makes use of a visual sensor. Using a visual sensor a least squares and an optical flow method have been deployed to estimate depth, that is the distance between a landmark and the robot. Both methods have low computational cost while at the same time they can produce robust depth estimates. It has been shown that there is a trade-off between accuracy of the estimates and high number of snapshots in the least squares approach, thus, decreasing any additional value of the optical flow method. Furthermore, optical flow outweighs least squares when the error in the least squares approach is large and the number of snapshots is small. A combination of the two approaches produces better depth estimation results.

The algorithms in this chapter provide the metric maps needed for the navigation of the robot. To support this method, a biologically-inspired model is presented in the next chapter where topological maps are built based on the optical flow ‘fingerprint’ of the landmarks. The following model is implemented with a single modality, that is a visual sensor. The requirements of the model to be presented are cost-efficient as only the properties of vectors comprising an image need to be stored. Furthermore, the only information used is optical flow. Velocity and position of the robot are considered during testing.

# Chapter 5

# Biological Maps

## 5.1 Introduction

This chapter presents the methods and results* obtained by applying a series of techniques for landmark visual navigation. Most of the methods applied have been inspired by biology. The work in this chapter implements some of the techniques described in Chapter 2. This chapter also deals with the problem of navigating a robot back to its home position using the methods discussed earlier. This chapter is concentrated on the biologically inspired algorithms while the previous one was focused on the engineering based algorithms.

The topological maps which are built by the robot are constructed during two different phases. In particular, when the robot departs from its home, it turns back towards home position and takes various snapshots of the area surrounding home. These snapshots are used for comparison when the robot has nearly finished the homing journey, i.e., when it is close to home location, and tries to locate its home position. The optic flow algorithm on the other hand, is used during the main course of both the foraging and homing journey. In other words, as soon as the Turn-Back-and-Look (TBL) approach has completed the robot starts building topological maps based on the 'fingerprint' of landmarks that is caused by the motion of the cameras which are perpendicular to the direction of motion. In this work translational optic flow is considered.

The first of these algorithms implements a strategy which was discussed in the second chapter when the template hypothesis was addressed. When an ant navigates, stores in its memory snapshots of prominent landmarks. Based on the size of the landmarks on their retina, ants are able to infer whether they have departed from or have come closer to the landmark compared to the position the initial snapshot was taken. A second technique that has been implemented is the detection of large-scale landmarks based on

*Figures in this chapter are best viewed in colour and most of them are placed at the end of the chapter.

the homogeneity of their colour. Detection of large-scale landmarks is required in this work in order to support the TBL phase.

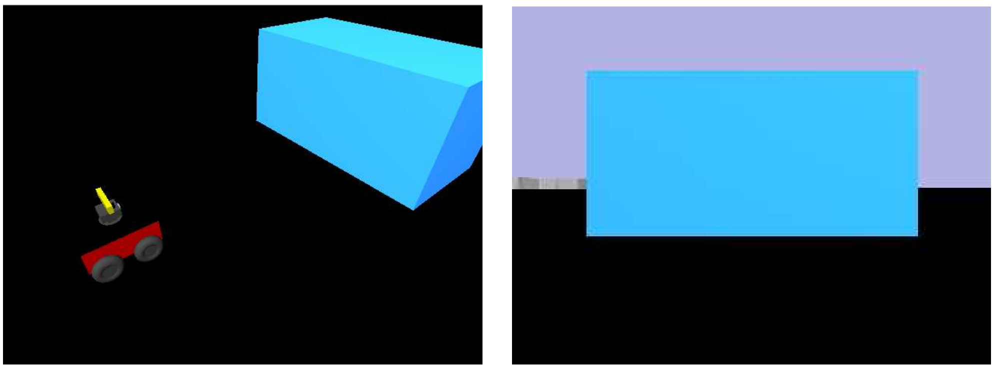
Furthermore, the TBL approach has been implemented for storing a series of different snapshots from different perspectives and varying illumination conditions. During the TBL process, the robot stores images of large-scale landmarks and their vicinity. The TBL approach is applied primarily at the beginning of the foraging journey since at that time the rate of change of landmarks is higher. For the detection and recognition of landmarks two block matching algorithms have been employed, namely the Sum of Absolute Differences (SAD) algorithm and the Normalised Cross Correlation (NCC) algorithm.

Finally, the last algorithm which has been developed is based on the optical flow algorithm of Lucas-Kanade. Using this algorithm and a training data set, the robot is able to infer a similarity score between the flow vectors of the current image and the flow vectors stored in the database. Storing only the vectors and not whole images is an important computational advantage. The robot, thus, tries to localise itself by calculating the similarity score based on the optical flow patterns of the current image and the patterns stored in the database. If the similarity score of vectors is high, then the robot recognises the location it is currently in otherwise it neglects it.

## 5.2 Retinal Size Landmark Simulation

As has been discussed in the second chapter, ants gauge distances from known landmarks by comparing their size in the retina at various locations. In this context, retinal size landmark simulation is used to support the TBL approach by inferring how far or close a landmark is. At the goal position an insect stores a global view of the scene, including some distinct landmarks. This process takes place when an insect has initiated its foraging trip. As soon as the insect has completed the outward journey the homing process begins. During its way back home the insect takes several snapshots of the environment and compares them with the stored initial image at the home position. So long as some landmarks are detected and recognised the insect takes an action on the basis of the retinal size of the landmarks. When a landmark is viewed from a distance its size on the retina is smaller, thus triggering an action towards the target. Conversely, when a landmark during the homing process looks larger compared to the stored initial image this implies that the insect has been closer to the landmark than the time it was captured during the foraging process.

The retinal size landmark technique has been implemented in the Gazebo (2006) simulator. Figures 5.1(a) and 5.1(b) show a snapshot of a landmark at the goal location.



(a) Bird's eye view of the environment

(b) Landmark view from robot camera

FIGURE 5.1: Snapshot of a landmark at goal location. Distance between robot and landmark is 2.5m

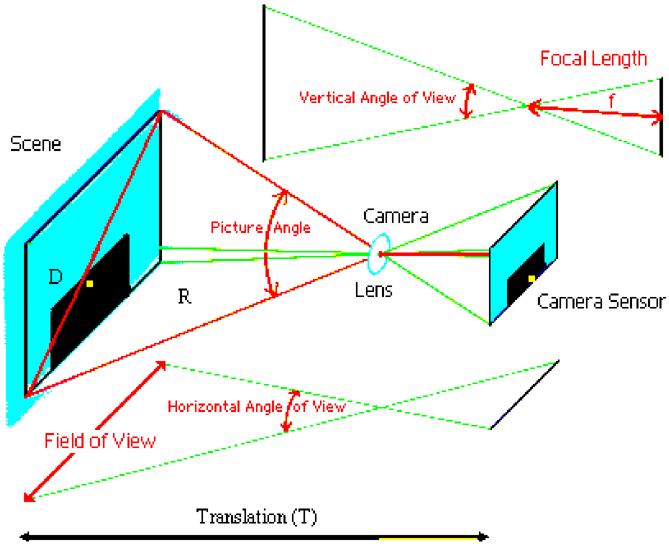
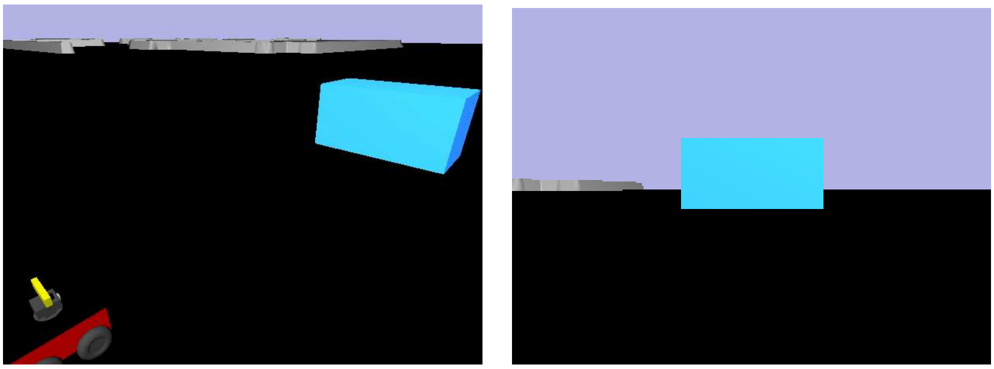


FIGURE 5.2: Camera and landmark parameters used for the simulation of landmarks in the retina.

In this simulation, the initial distance (or translation, $T_{x,y,z}$ ) of the robot to goal location is 2.5 metres, where $x = 2.5m$, $y = 0.0m$, and $z = 0.0m$. The rotation parameters (or Euler angles) $R_{r,p,y}$ (roll, pitch, yaw) have been kept constant to 0 radians. The dimensions $D_{x,y,y}$ of the landmark are $x = 1m$ (height), $y = 2m$ (width), and $z = 1m$ (depth). Finally, the known horizontal field-of-view (h-FOV) of the camera is 60°. Figure 5.2 shows the various parameters used for the simulation.

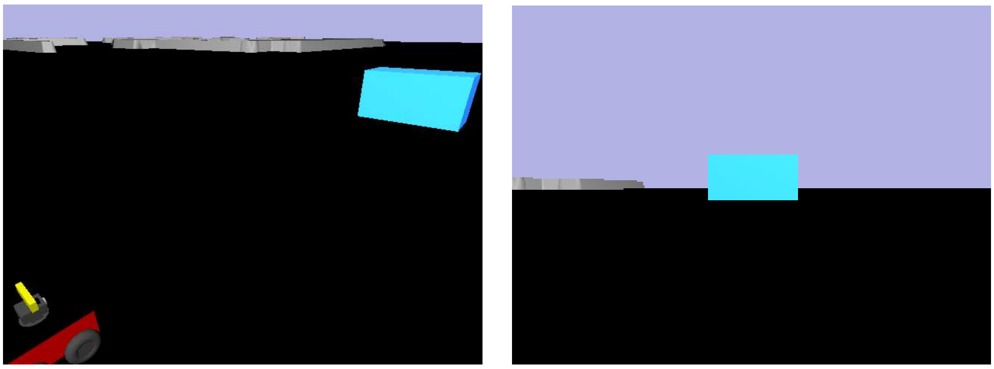
Furthermore, Fig. 5.3(a) and Fig. 5.3(b) show a bird's eye view of the environment and a snapshot of the robot camera respectively, at twice the distance from the goal location. In these figures, size, shape, and orientation of the landmark have been kept fixed while the translation $T_{x,y,z}$ has been increased by a factor of two of the initial location. Therefore, $x = 5.0m$, $y = 0.0m$, and $z = 0.0m$.



(a) Bird's eye view of the environment (b) Landmark view from robot camera

FIGURE 5.3: Snapshot at two times the distance from the goal location. Distance between robot and landmark is 5m.

Similarly, Fig. 5.4(a) and Fig. 5.4(b) show a bird's eye view of the environment and a snapshot of the robot camera, respectively. The distance between the robot and the camera is three times the distance of the initial snapshot, Fig. 5.1(b). The dimensions, $D_{x,y,z}$, of the landmark have been kept constant to the initial values, as in the previous figures, i.e., $x = 1m$, $y = 2m$, and $z = 1m$. Rotation $R_{r,p,y}$ has been kept, too, constant to 0 radians, while translation $T_{x,y,z}$ has the values $x = 7.5m$, $y = 0.0m$, and $z = 0.0m$.



(a) Bird's eye view of the environment (b) Landmark view from robot camera

FIGURE 5.4: Snapshot at three times the distance from the goal location. Distance between robot and landmark is 7.5m.

This information has been used in order to calculate the rate of change of a landmark's area with respect to distance. Figure 5.5 shows the rate of change of the area of a landmark as the distance between the robot and the landmark increases. The results shown in this figure, Fig. 5.5 will be used in the optical flow part of this work as the distance parameter will be considered when modelling distance in the training algorithm. In figure Fig 5.5 the slope of the function in the first few metres is higher than in the following ones. This reveals two things,

I. The rate of change of the landmark size in the retina is higher in the first few metres and is lowering as the distance increases.

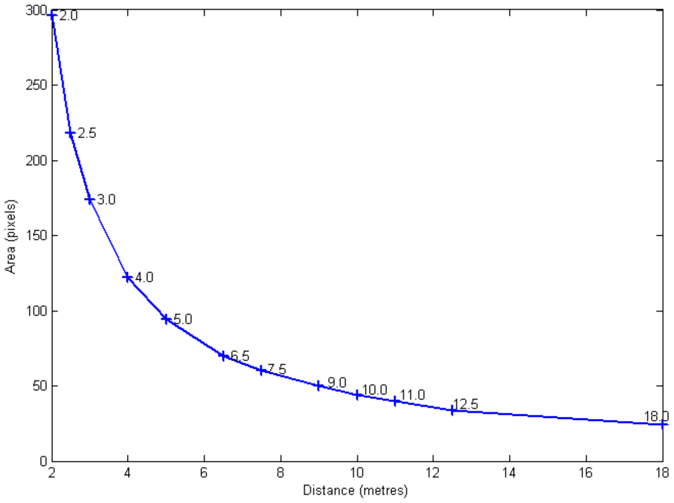


Figure 5.5: Rate of change of a landmark's area as a function of distance. The rate of change of a landmark's area decreases as the distance between the robot and the landmark increases.

II. The performance of calculating precisely the distance between a landmark and a camera drops as the size of the landmark decreases at a higher rate. This also reveals that error is proportional to the rate of change of the landmark size.

The finding that the rate of change of landmark size is increasing when approaching a landmark and, decreasing when departing from it is not a new one, but will prove useful when the turn-back-and-look approach is explained. Judd and Collett (1998) report that ants store multiple views of a landmark when they are close to it and the rate of change of its size in the retina continuously changes. Moreover, the number of snapshots taken for the same landmark has the tendency to decrease as its retinal size is stabilised.

## 5.3 Detection of Large-scale Landmarks

This section is devoted to an algorithm developed for the detection of large-scale landmarks. Various geometrical objects of varying size and colour have served as landmarks. The intensity of the pixels varies significantly, though they retain their colour over their whole surface. An illumination parameter has been added for this purpose, and the colour of the landmarks changes more abruptly as these are viewed from a larger distance.

The following algorithm calculates the Euclidean distance, $d$, between a reference pixel $R_f$ and the pixels, $C_j$, located on the circumference, i.e., on the boundaries, of the reference pixel. The purpose of this algorithm is to identify large and stable landmarks. The Euclidean distance has been employed as it can provide a metric distance between two points. It also requires less computation when compared to other distance measures like *Mahalanobis* distance. In the previous chapter it was shown how a landmark is tracked over time in order to infer the position of the robot. Furthermore, this algorithm as well as the one presented in the previous section, support the TBL approach by recognising the landmarks that surround the home position. The detection of large-scale landmarks is geared towards the global navigation scheme. For local purposes, other techniques can be applied such as the SIFT method proposed by Lowe (1999, 2004). SIFT features are invariant to scale, rotation, translation, and to some extent illumination changes in image space. The following equation, eqn. 5.1, shows how the large-scale detection algorithm works,

$$d = \sqrt{(r_R - c_R)^2 + (r_G - c_G)^2 + (r_B - c_B)^2} \tag{5.1}$$

where,

$$R_f = (r_R, r_G, r_B)$$

and,

$$C_j = (c_R, c_G, c_B)$$

with a radius of,

$$r = |R_x - C_x| \qquad r > 0$$

with $C_x$ and $R_x$ being the column coordinates of the template. Currently, the algorithm uses the $R, G, B$ colour space to compute the value differences between the reference pixel, $R_f$, and the pixels, $C_j$, located on the boundaries of a template. In order to compute the Euclidean distance, $d$, the radius, $r$, must be $r > 0$, i.e., the template must be odd-numbered and have a size of at least $3 \times 3$. Figure 5.6 providesa a pictorial representation of the algorithm.

The Euclidean distance, $d$, between the reference pixels and the pixels located in the circumference is compared to a given threshold, $T$. If any of the pixels of the circumference, $C$, of the template are above a threshold, $T$, then the reference pixel, $R$, is disregarded, otherwise it is considered a homogeneous pixel. Figures 5.7(a), 5.7(b), 5.7(c), and 5.7(d) demonstrate the input and output of the algorithm. The threshold value, $T$, that was used for this experiment was 100. Figures 5.8(a), 5.8(b), 5.8(c), and 5.8(d) demonstrate the results of the algorithm with lower resolution images and landmarks at smaller scales. A large landmark in this context refers to an object with a large

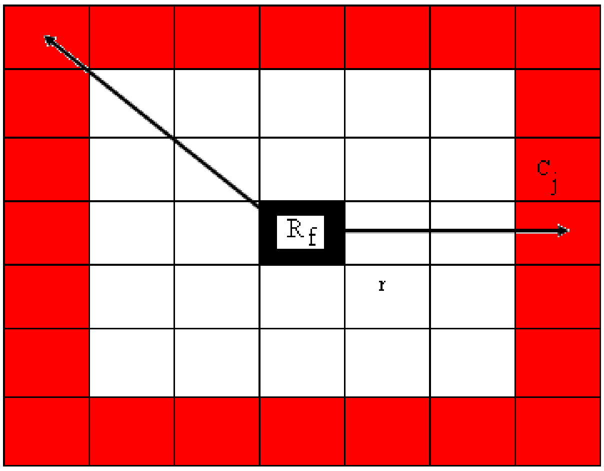
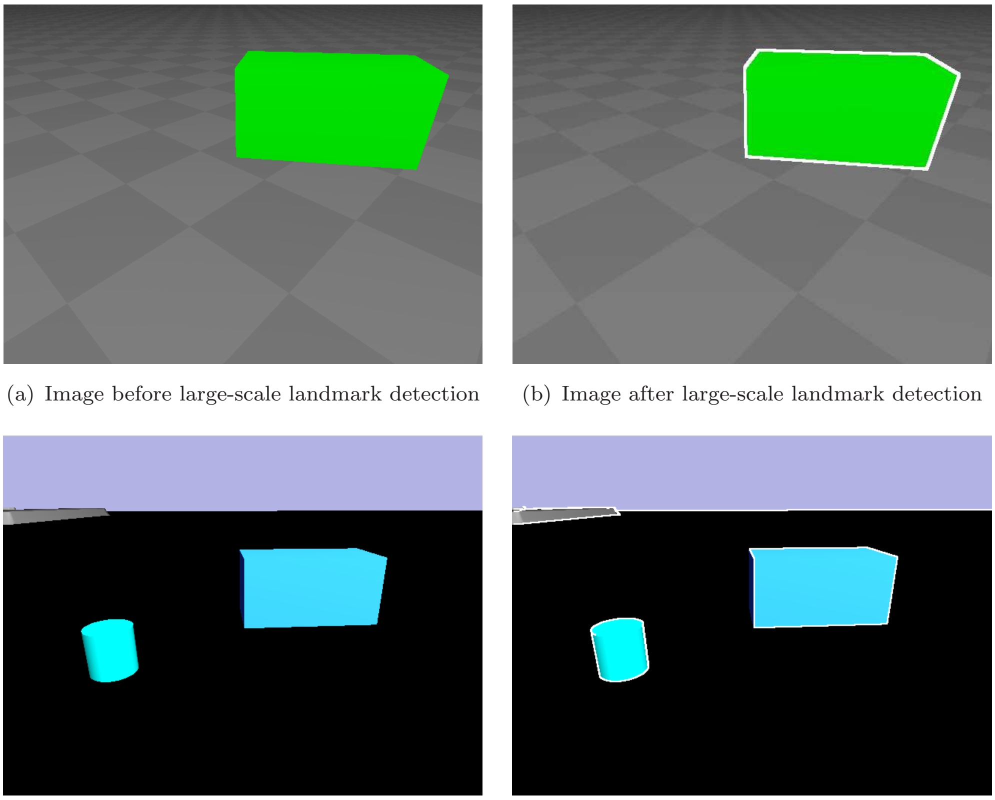


FIGURE 5.6: Reference pixel, $R_f$, in black cell, and boundary-pixels, $C_j$, in red cells, located in the circumference of the $7 \times 7$ template. Radius, $r$, in this example is 3.

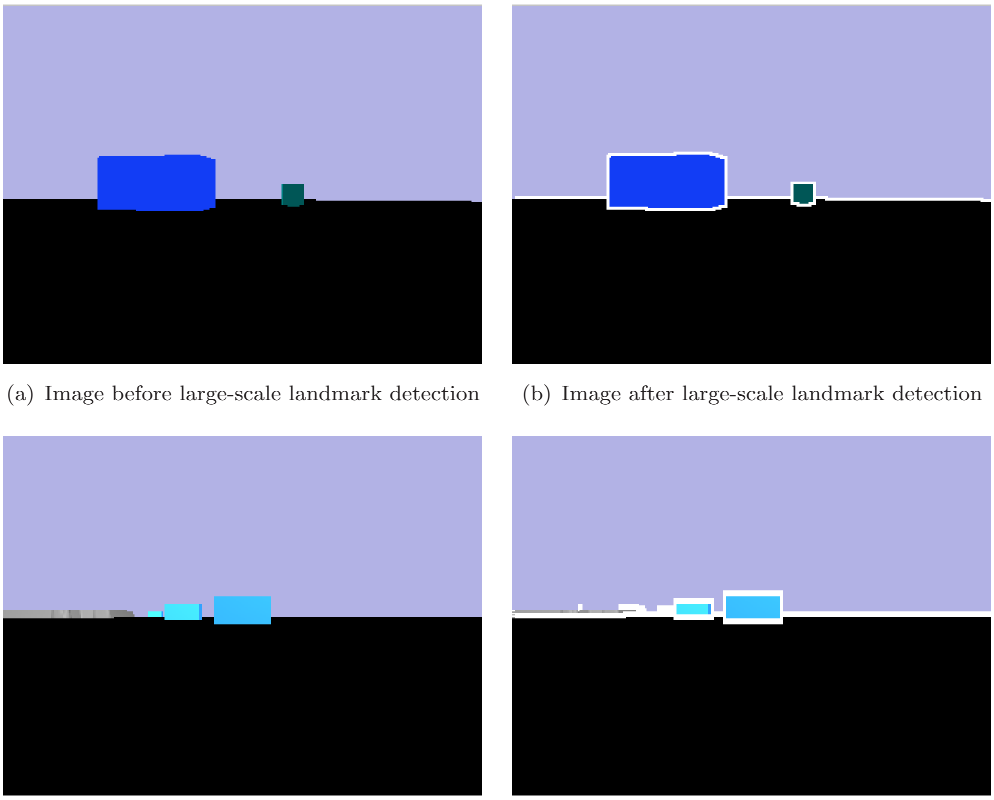


(a) Image before large-scale landmark detection (b) Image after large-scale landmark detection

(c) Image before large-scale landmark detection (d) Image after large-scale landmark detection

FIGURE 5.7: Large-scale landmarks detection. The resolution of images is 640 × 480.

amount of a homogeneous area. A landmark whose area does not fall within a specific bound then it is not considered a large landmark. A pattern image is thus extracted from a large scale landmark in order to match with an image. This algorithm is applied towards the end of the homing process in order to facilitate the robot locate its home position. Thus, metric distance is not considered at this step. Hence, a landmark that may look large due to its vicinity to the robot is not disregarded.



(a) Image before large-scale landmark detection

(b) Image after large-scale landmark detection

(c) Image before large-scale landmark detection

(d) Image after large-scale landmark detection

FIGURE 5.8: Large-scale landmarks detection. The resolution of images is 320 × 240.

The Euclidean distance algorithm for the detection of large-scale landmarks does not take into account pixels that fall within the template, only pixels that are situated on the circumference of it. From the few examples that the algorithm was tested with all the pixels that compose the template array, it was found that the algorithm performs quite inefficiently in terms of execution time. Nevertheless, the algorithm can find large homogeneous areas as the radius, $r$, increases. This, however, is at the expense of running time of the algorithm. One more parameter that affects the performance of the algorithm is the resolution of the image. The algorithm was tested for $640 \times 480$ and $320 \times 240$ images. A series of radii was used to test and evaluate the performance of the algorithm. The results are shown in Fig. 5.9.

## 5.4 Recognition of Large-scale Landmarks

This section describes and evaluates the performance of the two algorithms that were employed for the recognition of large-scale landmarks, namely the Sum-of-Absolute-Differences (SAD) and the Normalised Cross-Correlation (NCC). Both of these algorithms have been used extensively for visual navigation systems. SAD is simpler in

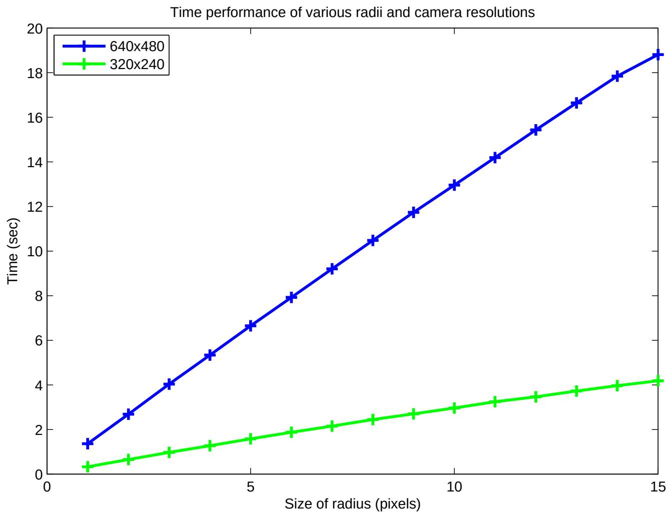


Figure 5.9: Time performance of various radii and camera resolutions. The tests have run on a Pentium 4 machine at 3.00 GHz with 1.00 GB of RAM.

complexity than NCC, but suffers from various parameters, such as lighting conditions, angle of view (or perspectiveness), and noise.

### 5.4.1 Sum of Absolute Differences algorithm

The Sum of Absolute Differences algorithm (SAD) used for area detection (or the recognition of the landmarks) is expressed by the following equation, eqn. 5.2,

$$SAD = \sum_{i=0}^{R-1} \sum_{j=0}^{C-1} |I_{ij} - T_{ij}|. \tag{5.2}$$

where $R$ is *rows*; $C$ is *columns*; and $I_{ij}$ and $T_{ij}$ represent the image and the template, respectively. The SAD algorithm employs a block matching process between a template image and an image for which the best match needs to be found. SAD algorithm subtracts the values of the corresponding pixels of the template and the image and the result is summed up until the whole search window is completed. The SAD algorithm produces a result of 0 when there is a perfect match between the image and the template. The larger the result of the correlation process, the less likely a matching occurs between the image and the template. The template image, $T_{ij}$, consists of a $3 \times 3$ array of the original image (5.1(b)), $I_{ij}$. The *z-axis*, or the correlation axis, has been normalised for the purposes of visualisation.

### 5.4.2 Normalised Cross-Correlation algorithm

Normalised Cross-Correlation is described by the following equation, eqn. 5.3,

$$NCC = \frac{\sum_{i=0}^{R-1}\sum_{j=0}^{C-1}(I_{ij}T_{ij})}{\sqrt{\sum_{i=0}^{R-1}\sum_{j=0}^{C-1}(I_{ij})^2 \cdot \sum_{i=0}^{R-1}\sum_{j=0}^{C-1}(T_{ij})^2}} \tag{5.3}$$

where $R$, represents the *rows*, and $C$ the *columns*. $I_{ij}$ and $T_{ij}$ represent the image and the template correspondingly. The range of a normalised cross-correlated matrix is $[-1..1]$, with 1 being a perfect match between the image $I_{ij}$ and the template image $T_{ij}$ (5.1(b)), and $-1$ for the opposite. A value greater than 0.8 is considered a good match. NCC involves more computations which are time consuming, like multiplications, divisions, and square roots. Banks and Corke (2001) present qualitative and quantitative comparison measures for various area detection algorithms, including the SAD and NCC algorithms, using images from stereo cameras. In this thesis, however, SAD and NCC algorithms are tested with different landmark scales and at varying distances. Yet, they have also been tested under different illumination conditions.

## 5.5 The Turn-Back-and-Look Approach

This section provides a means of comparison between the Sum of Absolute Differences (SAD) algorithm and the Normalised Cross-Correlation (NCC) algorithm using the turn-back-and-look behaviour. The turn-back-and-look, as discussed in Chapter 2, is a behaviour employed mainly by bees when depart from their hive.

Figure 5.10 demonstrates the trajectory of the robot while it retreats from its home performing the turn-back-and-look strategy. The arrows indicate camera poses. Yet, the multitude of arrows represents the number of snapshots taken, that is, the nearer to home position the more snapshots are taken. In the current section TBL is used for the comparison of SAD and NCC algorithms. Nevertheless, TBL has also been used for storing snapshots of the same landmark from different perspectives with the view to be used for matching in the homing journey of the robot. Lehrer and Bianco (2000); Bianco and Cassinis (2000), and Thompson et al. (1999) have used TBL for tracking landmarks as well as for selecting the most stable ones from a constellation of landmarks. TBL is a process that favours storing images whose rate of change in image space is changing rapidly. This occurs when the robot has initiated its outbound journey. During this period of time, the image space of the viewed landmarks is changing at a higher rate than in other instances.

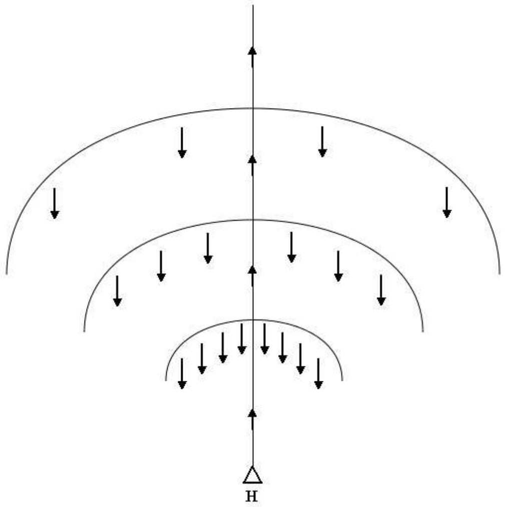


FIGURE 5.10: The turn-back-and-look approach. Arrows pointing towards hive (triangle) indicate camera poses.

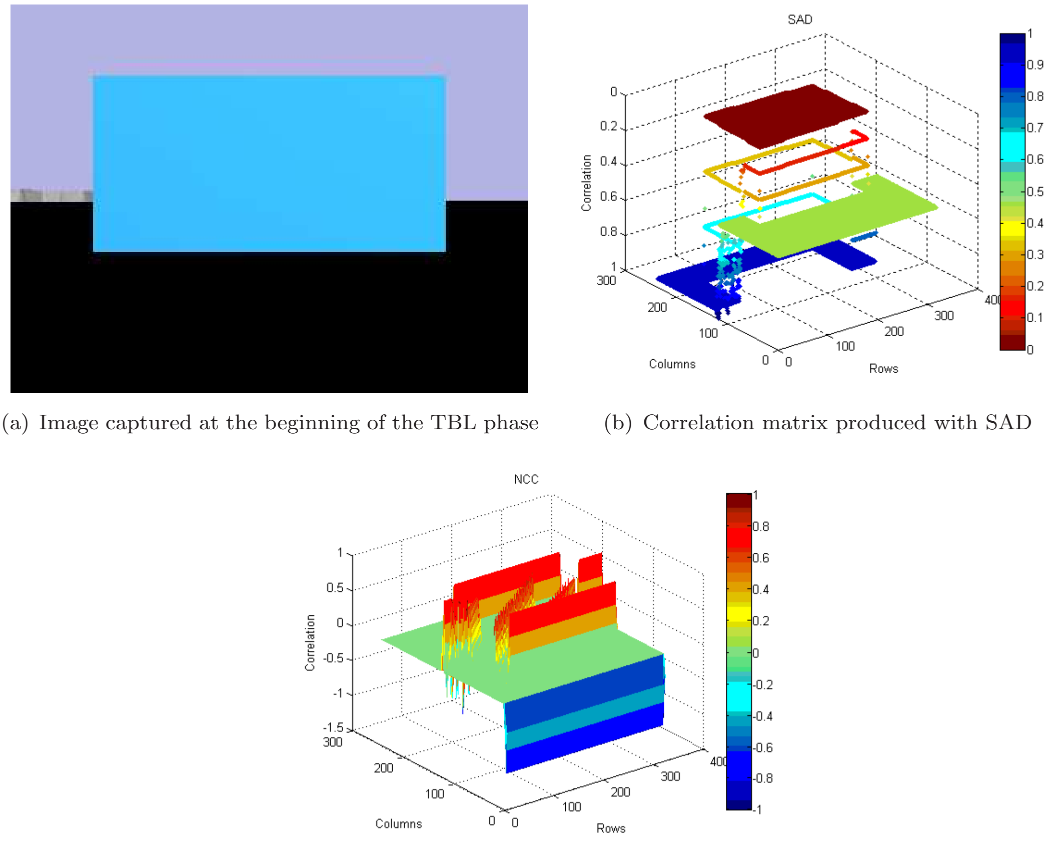
### 5.5.1 Landmark Recognition under Different Camera Perspectives

The figures in this section demonstrate the performance of the SAD and NCC algorithms employing the TBL strategy. The view of landmarks is now under different perspectives, i.e., different viewing angles and scaling. Figure 5.11(a) shows the landmark viewed at the initial stage of the TBL phase. The distance of the robot to the landmark is 2.5 metres. The dark red area reveals matching between a pattern image and the original image.

The same templates, as described earlier, have been used in all of the following images that implement the SAD and NCC algorithms. Figures 5.11(b) and 5.11(c) depict the correlation matrices of the SAD and NCC algorithms. In both images the matching occurred is at high levels. In Fig. 5.11(b) the dark red points reveal perfect matching. As the matching becomes less and less accurate, the colours change accordingly. Where there is no match, correlation matrix is of blue colour and, this is valid for both SAD and NCC correlation matrices. It should be noted that the images are transformed into grey level before the implementation of the algorithm.

The next figure, Fig. 5.26(a) has been taken at a distance of 5.0 metres between the robot and the landmark. The correlation matrix, Fig. 5.26(b), reveals that there is no absolute matching, but despite that, matching is quite high. The dark red square of Fig. 5.26(b) shows how much area of the landmark is close to exact matching. The correlation matrix, Fig. 5.26(c), of NCC algorithm shows too a small drop in matching the landmark with the image.

Figure 5.27(a) shows the landmark captured during the TBL phase while the robot is at an angular point of view. The correlation matrices, Fig. 5.27(b) and Fig. 5.27(c), of



(a) Image captured at the beginning of the TBL phase

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.11: SAD and NCC correlation matrices of landmark image at 2.5m distance (red colour depicts absolute matching in both graphs).

SAD and NCC correspondingly are similar to the previous ones with the NCC matching algorithm performing slightly better than SAD.

The next figure, Fig. 5.28(a), depicts the landmark at an even a larger translation of 7.5$m$. Once again, both the SAD and the NCC correlation matrices, Fig. 5.28(b) and Fig. 5.28(c) correspondingly, do not differ substantially with the previous correlation matrices. In fact, scaling seems not to be a major factor under the same illumination conditions.

Figure 5.29(a) is an image taken at an angular point of view, during the TBL approach, but the robot this time is moving left from the centre of the TBL motion. At this perspective the landmark is not homogeneously illuminated in all viewing sides, thus the dark-blue colour of the one side. The correlation matrices, Figs. 5.29(b) and 5.29(c), produced by the SAD and NCC algorithms correspondingly, reveal that SAD algorithm performs poorer that NCC whose correlation matrix remains almost identical to the previous one (Fig. 5.28(c)).

The next figure, Fig. 5.30(a), is again an image taken during the leftward-phase of TBL. The area of the landmark is decreasing as the robot retreats from its initial location.

The sides of the landmark are better illuminated compared to the previous example of Fig. 5.29(a). The correlation matrices of SAD and NCC, Fig. 5.30(b) and Fig. 5.30(c) correspondingly, differ significantly as NCC can achieve quite large amounts of matching. However, their matching has dropped even more because of the area reduction of the landmark.

Figure 5.31(a) shows the landmark at a larger distance from the initial location of the robot. The illumination is identical to the previous images with the difference that the scaling and rotation of the camera is larger. The viewing area of the landmark has become even smaller. The correlation matrices produced by SAD and NCC algorithms, Fig. 5.31(b) and Fig. 5.31(c) correspondingly, have dissimilar outputs as NCC outperforms SAD. The matching, however, has even more dropped since the landmark's area has decreased substantially.

The conclusion drawn from this section is that both SAD and NCC algorithms perform similarly when the area of the landmark remains stable and when illumination conditions do not change. When one or both of this parameters changes then NCC outperforms SAD. However, none of the algorithms result in an exact match so long as the TBL phase is initiated. The number of landmarks that are present in an image does not seem to influence their performance so long as matching occurs between the pattern and the image. Their performance drops as the TBL approach progresses and the robot retreats from its initial coordinates. The reason is obviously the decreasing area of the landmark viewed because of its increasing distance from the robot. The following section examines the two algorithms, SAD and NCC, under different illumination conditions.

### 5.5.2 Landmark Recognition under Different Illumination Conditions

This section demonstrates the performance of SAD and NCC algorithms using the TBL strategy under different illumination conditions. The first figure, Fig. 5.32(a), shows the same landmark as in Fig. 5.11(a) but with reversed lighting conditions and with the addition of one more landmark, a cylinder. The source of light in this image is in front of the robot camera causing landmarks to be viewed as dark-blue. The correlation matrix of SAD, Fig. 5.32(b), shows that SAD performs inadequately. The landmark's area is viewed in the correlation matrix in light green colour. The matching in that area of the matrix is about 0.65 with 0 being the perfect match. Certainly, it is the poorest result of the SAD algorithm encountered so far. The correlation matrix of NCC, Fig. 5.32(c), shows that NCC, in contrast to SAD, performs quite satisfactorily.

The following figure, Fig. 5.33(a), depicts a landmark from a close perspective and an angular point of view. The correlation matrix of SAD algorithm, shown in Fig. 5.33(b), reveals that the matching accuracy is higher this time. The illumination is different from the one of the previous section but the colour of the landmark remains largely identical to the initial image Fig. 5.11(a). The correlation matrix of NCC algorithm, Fig. 5.33(c), has a high degree of matching accuracy.

Figure 5.34(a) is similar to the previous image of Fig. 5.33(a) with the difference that the large side of the landmark is shadowy and is viewed by the robot camera as dark-blue. The results of the correlation matrices of SAD and NCC, Fig. 5.34(b) and Fig. 5.34(c) correspondingly, reveal that the SAD algorithm has performed a match in the area of the landmark whose side is not shadowy. In the dark-blue region of the landmark the matching has dropped. NCC's correlation matrix, on the contrast, performs approximately as good as in the previous image (Fig. 5.33(a)).

The image of Fig. 5.35(a) shows the landmark at perspectives with its sides being shadowed. The correlation matrix of SAD algorithm, Fig. 5.35(b), performs poorly since the landmark imprint is depicted on the correlation matrix with light green colour, meaning that its matching is at low levels. On the other hand, the correlation matrix of NCC algorithm, Fig. 5.35(c) performs satisfactorily, better than SAD, and its performance has remained stable.

The last image of the section, Fig. 5.36(a), shows the landmark at a close perspective with all its sides shadowed. The correlation matrix of SAD algorithm, Fig. 5.36(b), reveals that the matching occurred is quite poor, similar to the previous figure's, Fig. 5.35(a). NCC algorithm performs better in this shadowy image, too. Its correlation matrix, Fig. 5.36(c), retains a high degree of matching accuracy. From these tests, it can be concluded that SAD algorithm performs almost equally well with NCC when illumination remains constant, but it performs poorly when illumination within an image varies. The scaling parameter seems not to affect equally both algorithms. Although, at the beginning of the TBL process scaling did not seem to affect matching, it has, however, been shown that it becomes a drawback after a certain point. The matching accuracy of both algorithms at different scales is decreasing, but it still is high enough to perform a recognition task, especially for NCC. The rest of the chapter is devoted to the primary novelty of this chapter which is the localisation and topological map building using optic flow patterns.

## 5.6 Optic Flow-based Robot Localisation

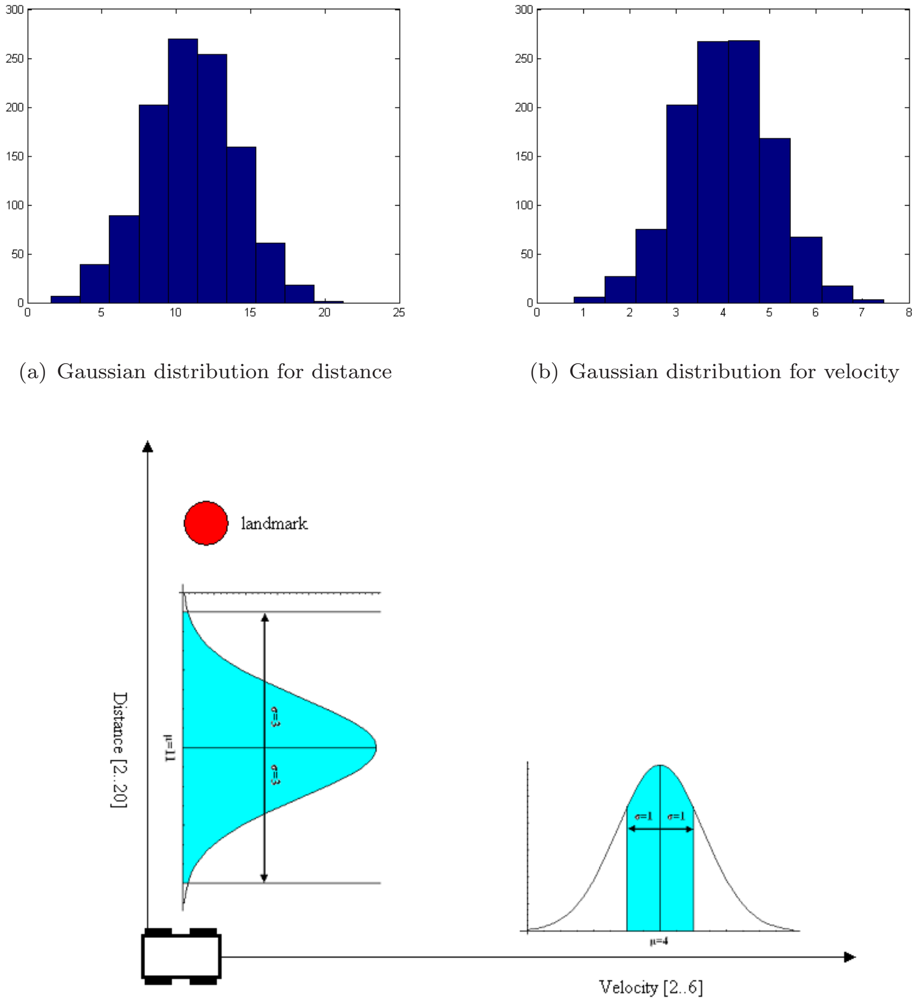
In this section the problem of localising a robot is addressed and is tackled using techniques found in biological organisms. Optical flow which has been used extensively lately for obstacle avoidance in this research has been employed to build topological maps of the environment. The purpose of this work is to show how optical flow can be used to create a *signature* of the environment based on the optical flow patterns perceived by the robot as it navigates through the environment. The only information used in this model is optical flow. Information such as position and velocity of the robot are not considered during navigation. Various landmarks have been modelled and simulated which the robot passes through. The simulated landmarks have geometrical shapes, like a sphere or a rectangular and they are textured (as is in real environments) in order to produce high optic flow. The simulated robot consists of two side-ways cameras which are perpendicular to the direction of motion. This creates a translational optic flow as the robot moves in the environment. Every landmark in the environment *emits* a number of optical flow vectors that are dependent on the distance of the robot to the landmark and the velocity of the robot. Comparing and storing only the properties of the vectors between different frames instead of the images greatly reduces the computation of the localisation process (Diamantas et al., 2010, In Press,b). The images are only captured and are not used for storage or comparison. The properties of the vectors refer to the number of vectors a given image possesses as well as the mean position of all the vectors in the image.

During the outbound trip of the robot the camera takes snapshots of the environment and using the Lucas-Kanade algorithm optical flow vectors are produced. During that phase the robot builds topological maps from the optical flow 'fingerprint' of the landmarks detected. After the foraging trip (or outbound trip), has completed, the homing trip (or inbound trip) is initiated. During the homing phase, the robot compares the optical flow patterns it currently perceives with the ones stored in the database, that is the ones created during the outward journey. If the similarity score between the two images is above a given threshold then the robot assumes the current landmark observed is the same with landmark observed during the outbound trip. The topological map of the robot is then used to localise the robot within an environment. The similarity score of the vectors is a probabilistic result of the Euclidean distance of the mean position of the vectors between the current image and the image stored in the database.

In order for the robot to localise in the environment using optical flow vectors, a training data set of $n = 1000$ observations has been implemented where a vector is observed at varying distances between the robot and the landmark, and at varying velocities. Thus, a joint probability distribution has been created by two continuous independent variables, that is velocity, $C$, distance, $D$, and is expressed by the following eqn. 5.4

$$f_{C,D}(c, d) = f_C(c) \cdot f_D(d) \qquad \forall c, d. \tag{5.4}$$

The velocity and the distance variables have been drawn from two Gaussian distributions with $\mu = 4, \sigma = 1$ and $\mu = 11, \sigma = 3$ respectively. The distances and velocities chosen to create the training data set approximate the real distributions of velocity and distance when a robot navigates in an environment. Figure 5.12(a) and Fig. 5.12(b) depict the histograms of the two variables, and Fig. 5.12(c) shows a pictorial representation of how the two variables, distance and velocity are modelled with respect to the robot. The $n = 1000$ observations model the position of the vector in the plane in a varying combination of distances and velocities. One assumption that needs to be met is that



(a) Gaussian distribution for distance

(b) Gaussian distribution for velocity

(c) Modelling of distance and velocity

FIGURE 5.12: The mean and standard deviation for distance (D) are $\mu = 11$, $\sigma = 3$, respectively, while for velocity (E) mean and standard deviation are $\mu = 4$, $\sigma = 1$, respectively. The range of the velocity modelled is from 2km/h to 6km/h and the range of distance modelled is from 2m to 20m. Figure 5.12(a) shows the Gaussian distribution used for modelling distance between robot and landmark. Figure 5.12(b) shows the Gaussian distribution of velocity that was used for modelling the velocity of the robot for navigating within an environment.

the majority of the vectors comprising a landmark should have the same or almost the same length. In order to solve the similarity problem between vectors, the mean, $x_k$, $y_k$, or centre point of every vector is taken. Thus, summing up all the mean points of the training data set and dividing by the number of observations the mean of the means is produced, $\bar{x}, \bar{y}$, as shown in eqn. 5.5

$$\bar{x}, \bar{y} = \frac{1}{n} \cdot \sum_{k=1}^{n} x_k, y_k \qquad n = 1000. \tag{5.5}$$

The mean of the means in an optic flow pattern can be visualised as the centre of gravity in a physical system. Another alternative to the mean of the mean could have been the matching of every single vector from the current snapshot with the vectors of the reference image. This, however, has proven infeasible as the number of vectors differ between the reference and the current image. Hence, the Euclidean distances, $\chi_k$, are computed between the mean of the means and the $n$ observations as expressed by eqn. 5.6

$$\chi_k = \sqrt{(x_k - \bar{x})^2 + (y_k - \bar{y})^2}. \tag{5.6}$$

The histogram produced by the Euclidean distances, $\chi_k$, of the training data set forms a *log-normal* probability density function (pdf) with $\mu = 2.24$ in the log location and $\sigma = 0.86$ in log scale. Figure 5.13(a) shows the histogram of vector deviations and the probability density function of the log-normal. The log-normal pdf is deployed in order to infer a similarity score (or probability as referred to in the graphs) as to how likely it is for the vectors of the current snapshot to have deviated when compared with the vectors of a snapshot stored in memory. In addition, the log-logistic probability density function could have been used as it equally models well the distribution of the vector deviations. Nevertheless, log-normal is believed to behave better than log-logistic for larger data sets. Figure 5.13(b) depicts the cumulative distribution function (cdf) of vector deviations and the lognormal.

The probability density function of log-normal is given by the eqn. 5.7,

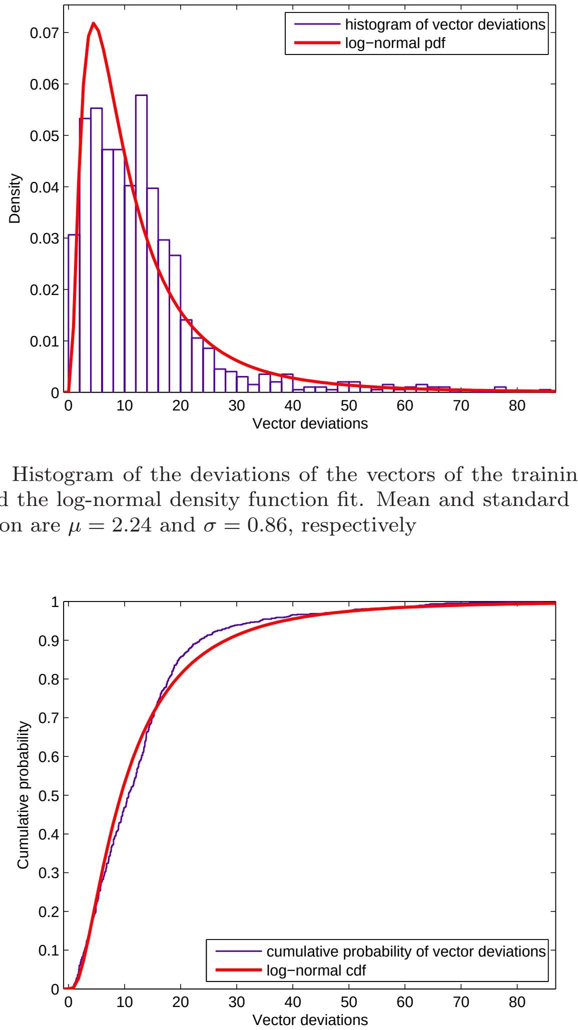
$$f_X(\delta; \mu, \sigma) = \frac{1}{\delta\sigma\sqrt{2\pi}} e^{-\frac{(\ln\delta - \mu)^2}{2\sigma^2}} \qquad x > 0 \tag{5.7}$$

and the cumulative density function of log-normal is expressed by eqn. 5.8,

$$F_X(\delta; \mu, \sigma) = \frac{1}{2} erfc\left[-\frac{\ln\delta - \mu}{\sigma\sqrt{2}}\right] = \Phi\left(\frac{\ln\delta - \mu}{\sigma}\right) \tag{5.8}$$

where *erfc* is the complementary error function and $\Phi$ is the standard normal cdf. During the homing navigation process, the robot calculates the Euclidean distance, $\delta$, between the mean position, $\bar{x}, \bar{y}$, of all the vectors in a given landmark with the mean position of the vectors of the landmarks stored in the database. Equations 5.9, 5.10, and 5.11 describe the process for two distinct landmarks. In eqn. 5.9, and eqn. 5.10, $r$ and $s$ are the number of vectors for two distinct landmarks $i$ and $j$, one of which is observed during the outbound trip while the other one is observed during the inbound trip.

$$\bar{x_i}, \bar{y_i} = \frac{1}{r} \cdot \sum_{a=1}^{r} x_a, y_a \tag{5.9}$$



(a) Histogram of the deviations of the vectors of the training set and the log-normal density function fit. Mean and standard deviation are $\mu = 2.24$ and $\sigma = 0.86$, respectively

(b) Cumulative density functions of the deviations of the vectors and the log-normal distribution

FIGURE 5.13: Probability density function and cumulative density function of vector deviations and the log-normal distribution.

$$\bar{x_j}, \bar{y_j} = \frac{1}{s} \cdot \sum_{b=1}^{s} x_b, y_b \tag{5.10}$$

$$\delta = \sqrt{(\bar{x_i} - \bar{x_j})^2 + (\bar{y_i} - \bar{y_j})^2} \tag{5.11}$$

$$P = 1 - P_\delta \tag{5.12}$$

The log-normal cdf then gives us the probability $P_\delta$ based on the Euclidean distance $\delta$ between the two sets of vectors. It is then subtracted from 1 to give the probability $P$ as is in eqn. 5.12. In addition, the probability $P$ of the log-normal is multiplied by the ratio of the number of the vectors as shown in eqn. 5.13 with $min_i$ being the landmark $i$ with the minimum number of vectors and $max_j$ being the landmark $j$ with the maximum number of vectors.

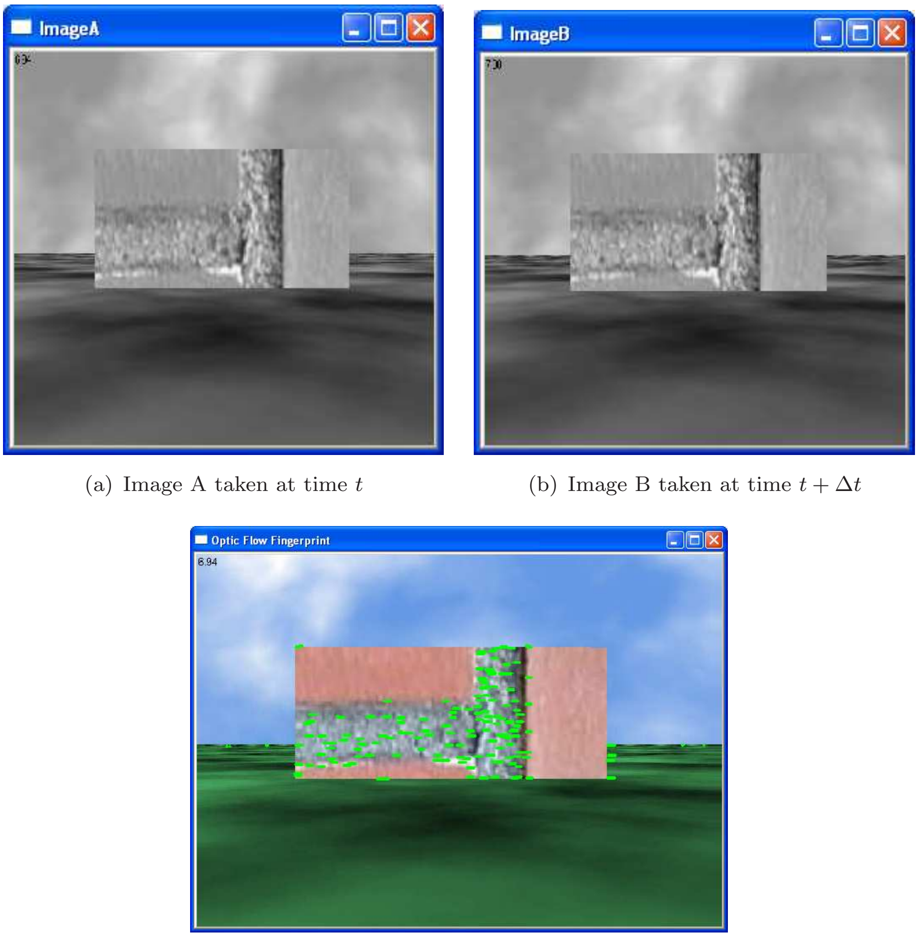
$$P_T = P\left(\frac{min_i}{max_j}\right) \tag{5.13}$$

This results to the total probability (or similarity score), $P_T$. Thus, even if the Euclidean distance, $\delta$, between two sets of vectors is small, the total probability, $P_T$, can be low if the ratio of the vectors is small. Hence, two patterns which are totally different may have a small Euclidean distance that yields a high probability. Multiplying the probability value, $P$, by the ratio of the number of vectors can drop significantly the total probability value, $P_T$, assuming that the number of vectors of the two sets are not of the same multitude.

Figure 5.14(c) shows the optical flow pattern of the landmark taken at a distance of 11m and with a velocity of 4km/h. The optical flow images are produced by calculating the motion of an entity at two contiguous frames. It should also be noted that the flow vectors appear upside down since the images are read upside down, too. Figure 5.14(a) and Fig. 5.14(b) show the first and the second frame in sequence, correspondingly. The landmark of Fig. 5.14(c) acts as a reference landmark for the following snapshots in order to demonstrate the similarity score at varying distances and velocities, and between different landmarks.

In this work a clustering algorithm has also been implemented for counting the number of landmarks. The counting of landmarks is a method that is used by insects as mentioned in Chapter 2. Although the clustering algorithm is a rather simple one, distinguishing between different landmarks is an important task. The number of landmarks each one of the two cameras can distinguish is two. Therefore four is the maximum number of landmarks that can be seen at any one time. The matching process is independent for each camera. Thus, in the case where one or more landmarks have gone missing then the robot can recognise its location from the landmarks which are still visible in the other camera. The clustering algorithm creates a histogram of the vectors and finds the minimum point between the two peaks of the histogram. Figure 5.15 shows an application of the algorithm with two landmarks. Vectors with small length, that is equal to or less than 1 are considered as outliers. In the same figure a number of outliers has been detected.

The following figures explain the effectiveness of the model to localise a robot within an environment based on the optical flow patterns of the landmarks. The first set of figures Figs. 5.16(a)-5.16(c) and Figs. 5.17(a)-5.17(c) show the optical flow patterns between the reference image, Fig. 5.14(c), and the patterns of the same image taken at different time steps, $t$. The circles in the graphs represent the mean position of all the vectors that comprise a given landmark and deviation refers to the Euclidean distance between the centres of the circles. The *Green* colour refers to the vectors of the reference image



(a) Image A taken at time $t$

(b) Image B taken at time $t + \Delta t$

(c) Optic flow

Figure 5.14: Snapshot of a reference landmark taken at a distance of 11m and a velocity of 4km/h

while *Blue* colour refers to the current image. The current snapshot has been taken at a distance of 8m and a velocity of 4km/h. The reference image has been taken at a distance of 11m and a velocity of 4km/h. As can be seen from the figures, the difference in the similarity score occurs due to the different position the snapshots were taken. The highest similarity score, 45.01%, is achieved in Fig. 5.16(a). In the same figure the Euclidean distance between the two images is the smallest among the figures of the set. In the remaining figures, Figs. 5.16(b)-5.16(c), it can be seen that deviation increases as the time step, $t$, changes. Another feature that is affected by the different time step an image has been captured is the number of elements. In Fig. 5.16(a) the number of elements is 308. The number of elements in the neighbouring snapshots drop or increase according to whether they have been taken before or after Fig. 5.16(a). In the last two figures of the set, that is Fig. 5.17(b) and Fig. 5.17(c) the similarity score is low revealing that the difference in time step an image is captured can be critical to the localisation process.

In the next set of figures, that is Figs. 5.18(a)-5.18(c) and Figs. 5.19(a)-5.19(c) the

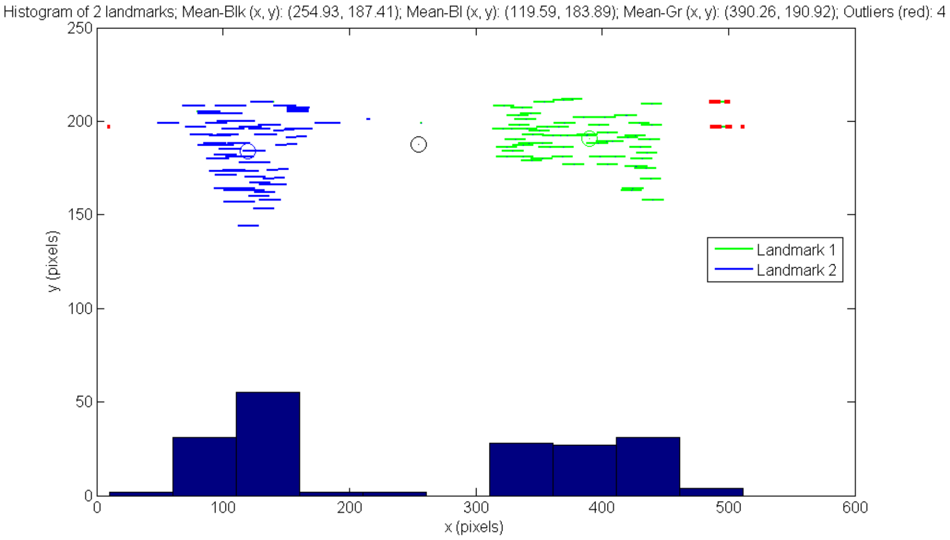


Figure 5.15: Clustering of two landmarks by employing the histogram of their vectors. The circles which are located within optical flow vectors depict the mean position of the corresponding optical flow patterns. The circle in the centre of the graph depicts the minimum point of the histogram.

similarity score, 92.71%, is at the highest level, approaching almost the perfect match, Fig. 5.18(a). In this set of figures the distance has been kept the same as in the reference image, that is 5m, but changing the velocity from 4km/h to 5km/h. The similarity scores in four out of the six snapshots of the set is quite high. However, the snapshots of Fig. 5.19(b) and Fig. 5.19(c) give a low similarity score. These snapshots have been taken at $t = t - 2\Delta t$ and $t = t - 3\Delta t$, respectively which shows why the low similarity score occurs. Deviation in these two snapshots is also quite large, that is 30.28 and 40.23 for Fig. 5.19(b) and Fig. 5.19(c), respectively. From this simulation it can be concluded that distance affects more than velocity the optical flow patterns.

In Figs. 5.20(a)-5.20(c) and Figs. 5.21(a)-5.21(c) both distance and velocity parameters have changed to 8m and 5km/h, respectively. The similarity score of 30.61% (Fig. 5.20(a)) is rather sensible if we take into consideration that the similarity score of the same landmark at the same distance and a velocity equal to 4km/h was 45.01%, Fig. 5.16(a). In the same figure, Fig. 5.20(a), deviation is small when compared to the rest of figures in the same set. The remaining figures depict a drop in the similarity score as $t$ changes. The larger the $\Delta t$ the smaller the similarity score achieved. Tables 5.1-5.3 summarise the results of the optical flow method at varying distances and velocities. The similarity scores refer to the same reference landmark taken at different time steps and at varying velocities and distances.

In the next set of images the similarity score is inferred between the reference image of Fig. 5.14(c) and a mountain-like landmark, Fig. 5.22. Figures 5.23(a)-5.23(d) depict the optical flow patterns at varying distances and velocities. In Fig. 5.23(a) the distance and

Table 5.1: Optical Flow Performance at $8m$ and $4km/h$

| Time $t$ | No of vectors | Deviation | Probability (%) |
|---|---|---|---|
| $t+3\Delta t$ | 394 | 24.58 | 7.71 |
| $t+2\Delta t$ | 343 | 25.32 | 8.37 |
| $t-2\Delta t$ | 294 | 14.33 | 24.63 |
| $t+\Delta t$ | 395 | 10.67 | 26.01 |
| $t-\Delta t$ | 306 | 8.07 | 43.50 |
| $t$ | 308 | 7.66 | 45.01 |

Table 5.2: Optical Flow Performance at $11m$ and $5km/h$

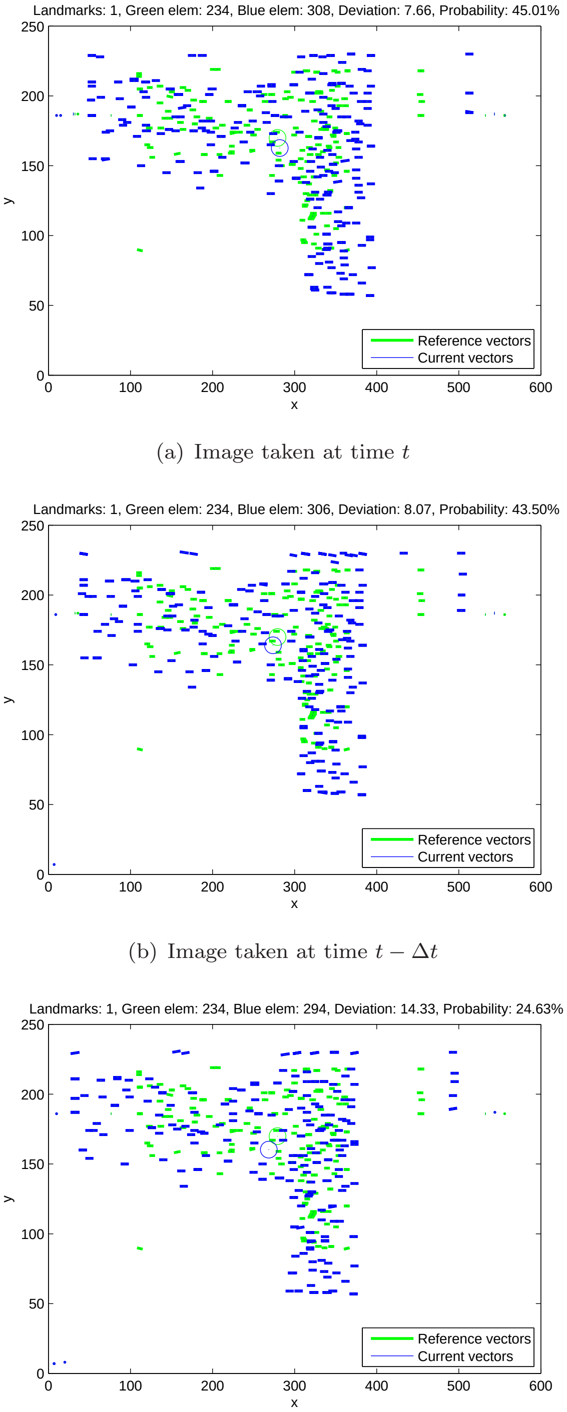
| Time $t$ | No of vectors | Deviation | Probability (%) |
|---|---|---|---|
| $t-3\Delta t$ | 265 | 40.23 | 3.92 |
| $t-2\Delta t$ | 273 | 30.28 | 7.31 |
| $t+2\Delta t$ | 237 | 13.42 | 33.27 |
| $t-\Delta t$ | 203 | 9.52 | 42.70 |
| $t+\Delta t$ | 219 | 5.21 | 70.49 |
| $t$ | 227 | 2.18 | 92.71 |

Table 5.3: Optical Flow Performance at $8m$ and $5km/h$

| Time $t$ | No of vectors | Deviation | Probability (%) |
|---|---|---|---|
| $t+4\Delta t$ | 223 | 47.05 | 2.83 |
| $t+3\Delta t$ | 198 | 33.97 | 5.60 |
| $t+2\Delta t$ | 205 | 27.63 | 9.04 |
| $t-\Delta t$ | 254 | 21.64 | 15.11 |
| $t+\Delta t$ | 169 | 14.18 | 22.67 |
| $t$ | 174 | 11.33 | 30.61 |

velocity parameters have been kept the same as in the reference image. The similarity score, 7.26%, is very low revealing that the optical flow patterns are dissimilar. A similar probabilistic score is inferred in Fig. 5.23(c) as well. In the remaining figures, that is Fig. 5.23(b) and Fig. 5.23(d) the similarity score calculated is even lower that that of the previous images. In Fig. 5.23(b) the similarity score is 2.29% and deviation between the current image and the reference image is 44.19 while in Fig. 5.23(d) the similarity score is 1.63% and deviation is 49.93. The number of elements in the current images differ substantially with the number of elements in the reference image contributing even more to the low probabilistic scores. From this last set of figures it can be seen that similarity score cannot achieve acceptable levels even if the distance and velocity parameters remain the same as in the reference snapshot. A low similarity score is not regarded by the robot while a high similarity score is used to localise the robot with respect to the topological map.

Figure 5.24(a) depicts another landmark whose optical flow pattern, Fig. 5.24(b), is compared with the reference's landmark, Fig. 5.14(c). The distance and velocity the image was taken are the same as in reference image, that is 11m and 4km/h, respectively.

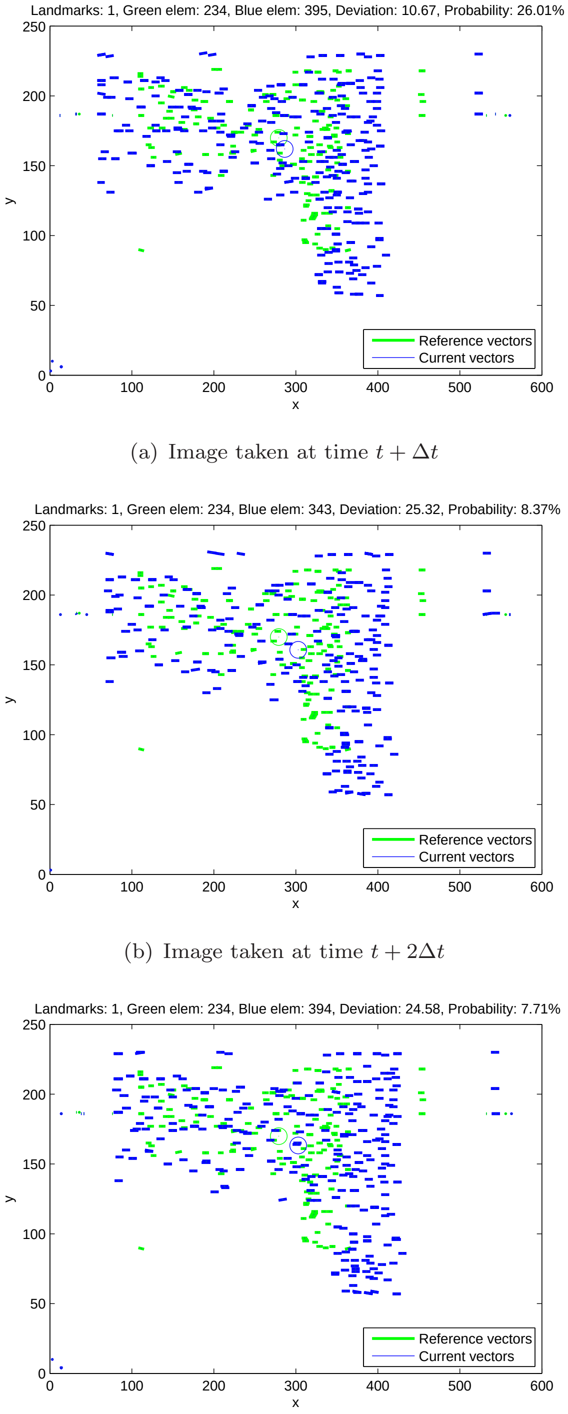


(a) Image taken at time $t$

(b) Image taken at time $t - \Delta t$

(c) Image taken at time $t - 2\Delta t$

FIGURE 5.16: Snapshot of reference landmark taken at a distance of 8m and a velocity of 4km/h. In these graphs, the reference landmark Fig. 5.14(c) which has been taken at a distance of 11m and a velocity of 4km/h is compared against the same landmark taken at a different distance and velocity. Similarity score (or Probability) decreases as the time step a snapshot was taken differs from first one, Fig. 5.16(a). This is reflected in the increasing deviation (Euclidean distance) between the two optical flow patterns. The circles in the graphs depict the mean position of the corresponding optical flow vectors. (A).



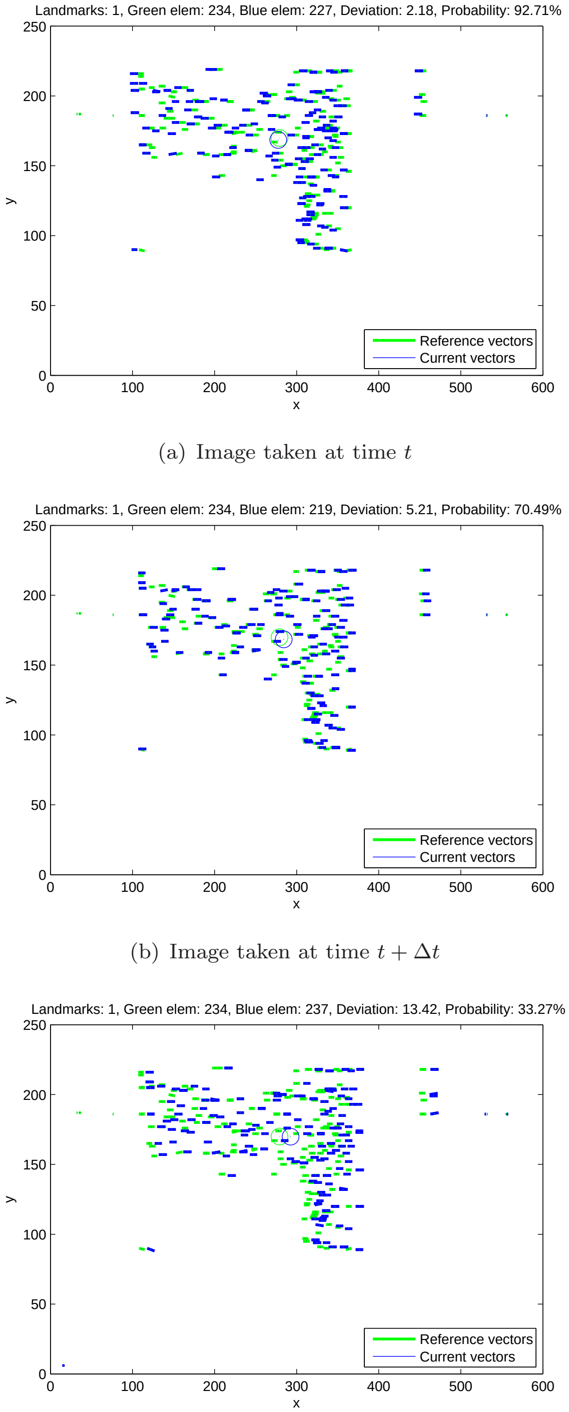
(a) Image taken at time $t + \Delta t$

(b) Image taken at time $t + 2\Delta t$

(c) Image taken at time $t + 3\Delta t$

FIGURE 5.17: Snapshot of reference landmark taken at a distance of 8m and a velocity of 4km/h. Similarity score (or Probability) decreases as the time step a snapshot was taken differs from first one, Fig. 5.16(a). The circles in the graphs depict the mean position of the corresponding optical flow vectors. (B).

The similarity score inferred by the patterns of the two images is quite low, that is 2.04% revealing that the two snapshots are different to each other. This is a result of the large deviation between the two graphs, that is 44.40. The low ratio of the number

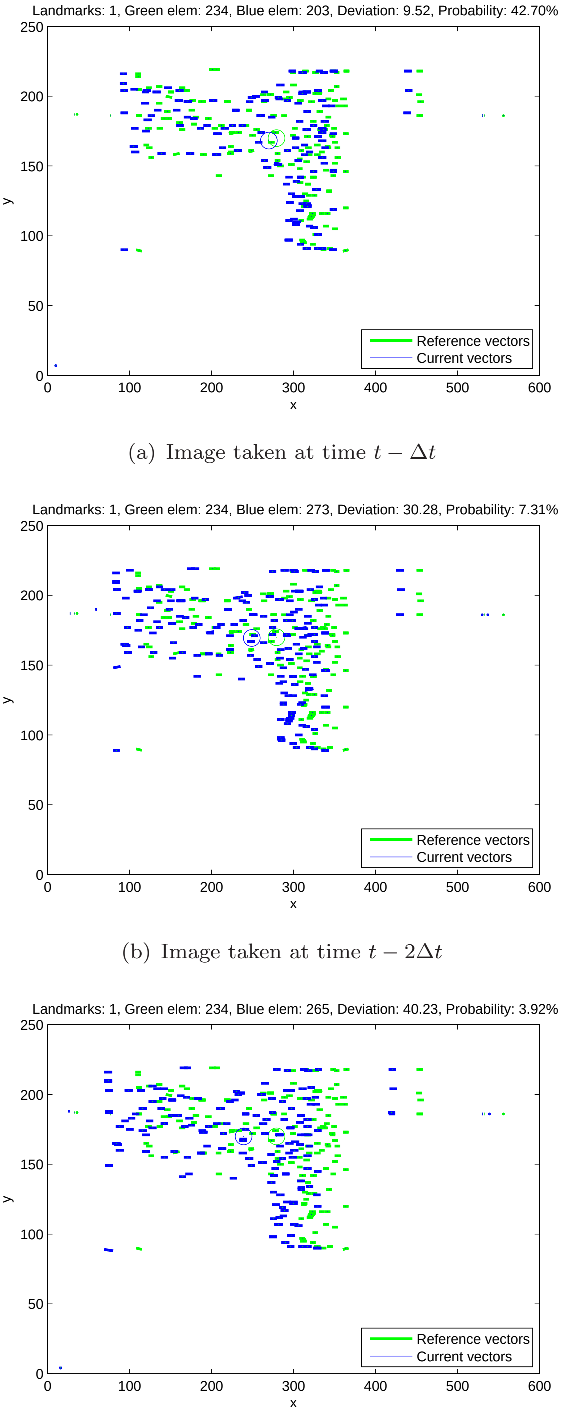


(a) Image taken at time $t$

(b) Image taken at time $t + \Delta t$

(c) Image taken at time $t + 2\Delta t$

FIGURE 5.18: Snapshot of reference landmark taken at a distance of 11m and a velocity of 5km/h. Reference landmark of Fig. 5.14(c) is compared against the same landmark but at a distance of 11m and a velocity of 5km/h. Figure 5.18(a) shows that a high similarity score is achieved between the two optical flow patterns. Velocity, therefore, is less influential that distance. Similarity score is high even when time steps differ, Fig. 5.18(b) and Fig. 5.18(c). The circles in the graphs depict the mean position of the corresponding optical flow vectors. (A)



(a) Image taken at time $t - \Delta t$

(b) Image taken at time $t - 2\Delta t$

(c) Image taken at time $t - 3\Delta t$

FIGURE 5.19: Snapshot of reference landmark taken at a distance of 11m and a velocity of 5km/h. Similarity score drops further as the time step increases. The circles in the graphs depict the mean position of the corresponding optical flow vectors. (B).

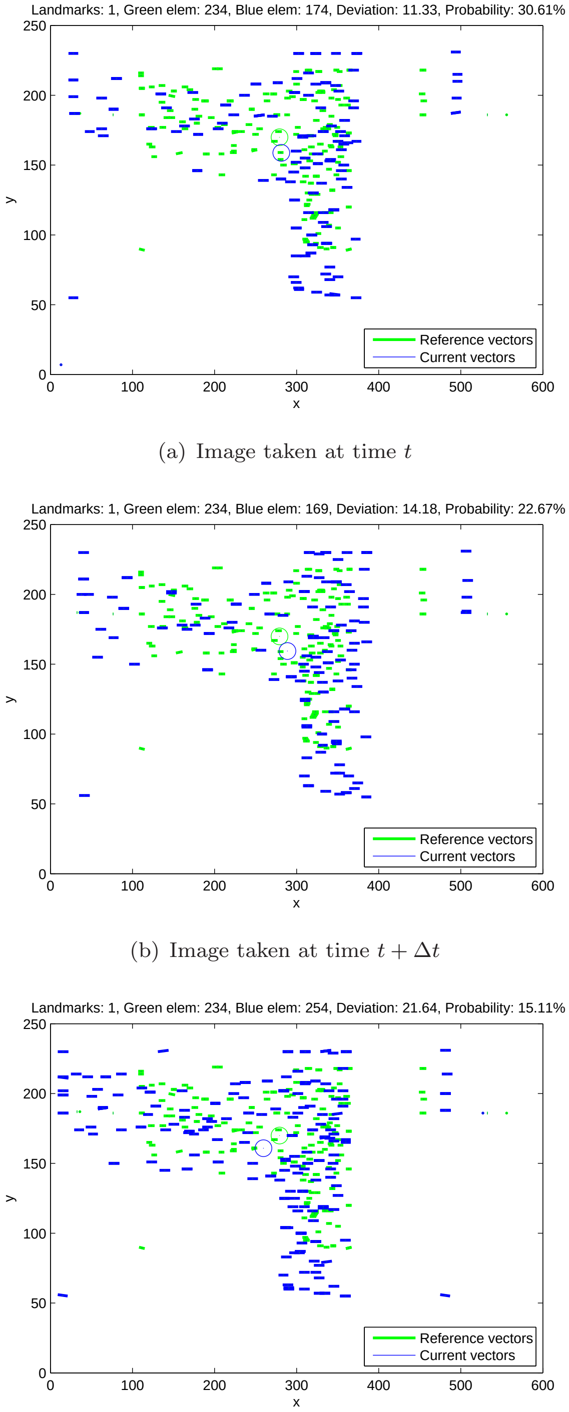
of elements of the two graphs has also contributed to the low similarity score. The landmark of Fig. 5.24(c) is also compared with the reference landmark of Fig. 5.14(c). This parallelogram landmark is similar to the reference one, in terms of shape, size, and texture. The optical flow pattern of the landmark is depicted in the next figure, Fig. 5.24(d) where the similarity score achieved is 39.13%. Although the ratio of the number of their elements is small the deviation between the patterns has caused a low probabilistic score between the patterns. This is a case where the two patterns refer to different landmarks but the similarity score is high enough to be considered as identical landmarks. In such a scenario, the robot will consider as having revisited the place, thus leading to erroneous actions on behalf of the robot.

Figure 5.25(a) depicts two landmarks in one snapshot and their corresponding optical flow patterns in Fig. 5.25(b). At the reference image of Fig. 5.25(b) (shown in *Green* for the tower and *Red* for the sphere) the distance between the robot and the sphere is 5m and, 11m between the robot and the tower. The velocity the image was taken is 4km/h. The current snapshot of Fig. 5.25(b) (shown in *Blue* for the tower and *Black* for the sphere) has been taken 2m further from the reference landmark, that is 7m from the sphere and 13m from the tower. The velocity has been increased to 5km/h. The similarity score for the tower landmark is 22.55% and for the sphere is 27.08%. This is an important finding since the two similarity scores do not differ significantly. The distance between the current and the reference image has increased by 2m but the distance between the two landmarks is 6m. This shows that it does not matter how far the reference image was taken so long as the distance between the reference and current image does not differ significantly. In the next Fig. 5.25(c) two different landmarks are shown. These two landmarks are compared against the landmarks of the previous image, Fig. 5.25(a). Their optical flow patterns appear in Fig. 5.25(d). The distance between the robot and the mountain landmark is 8m while the distance between the robot and the tower landmark is 11m. The velocity the image was taken is 3km/h. The similarity score between Fig. 5.25(a) and Fig. 5.25(c) is quite low showing that the two patterns are different to each other. In particular, the similarity score between the two towers is 8.34% while the similarity score between the sphere and the mountain is 0.93%. The similarity score is higher between the two towers as their shape and size resemble to each other. From the simulations it can be concluded that a similarity score of a landmark above 20% is considered satisfactory to be used for the localisation of the robot. There are, however, cases as shown earlier, where a landmark may falsely be recognised and thus be used for localisation.

## 5.7 Conclusion

This chapter has presented the biologically-inspired methods for tackling the homing problem. At first, a detection algorithm was presented that detects large-scale and homogeneous landmarks in the environment. This method is used at the beginning of the outbound trip, that is, during the Turn-Back-and-Look approach. At this point the robot turns towards its home position and takes snapshots of the area surrounding its home. The number of snapshots taken decreases as the robot moves outwards. The reason for this is that the rate of change in the size of the landmarks decreases with distance. The snapshots of the home position are taken at different positions and viewing angles. These snapshots will be used by the robot at the end of the homing process when robot is close to its home position and the precise position of its home location needs to be found. For recognising the landmarks at the end of the homing process two different area detection algorithms have been tested, namely Sum of Absolute Differences and Normalised Cross Correlation. The later one seems to outperform the former one when an image is taken at different viewing angles and distances.

During the main course of the navigation, that is, when the robot has completed its Turn-Back-and-Look approach and has initiated its outward journey the robot creates optical flow patterns from images. From these optical flow patterns only the mean position of all the vectors and the number of vectors are stored. This greatly reduces the computational complexity of the model in contrast to traditional visual navigation methods where whole images need to be stored and retrieved. In fact, the only information used in this model is optical flow. Position and velocity of the robot are not known during testing. They are only modelled in the training algorithm. An additional novelty of the model is that optical flow is not a property of the landmarks such as colour, shape, and size but a property of the camera motion. When the robot has completed its outward journey and homing has initiated, the optical flow patterns perceived by the cameras of the robot are compared with the patterns stored in memory. A similarity score is inferred between the two patterns. If the similarity score is above a given threshold, as shown by simulations above 20.0% is satisfactory, then the robot localises itself with respect to the environment and a topological map is created based on the optical flow patterns perceived. The following chapter presents the conclusions drawn from this work and a future work is addressed.

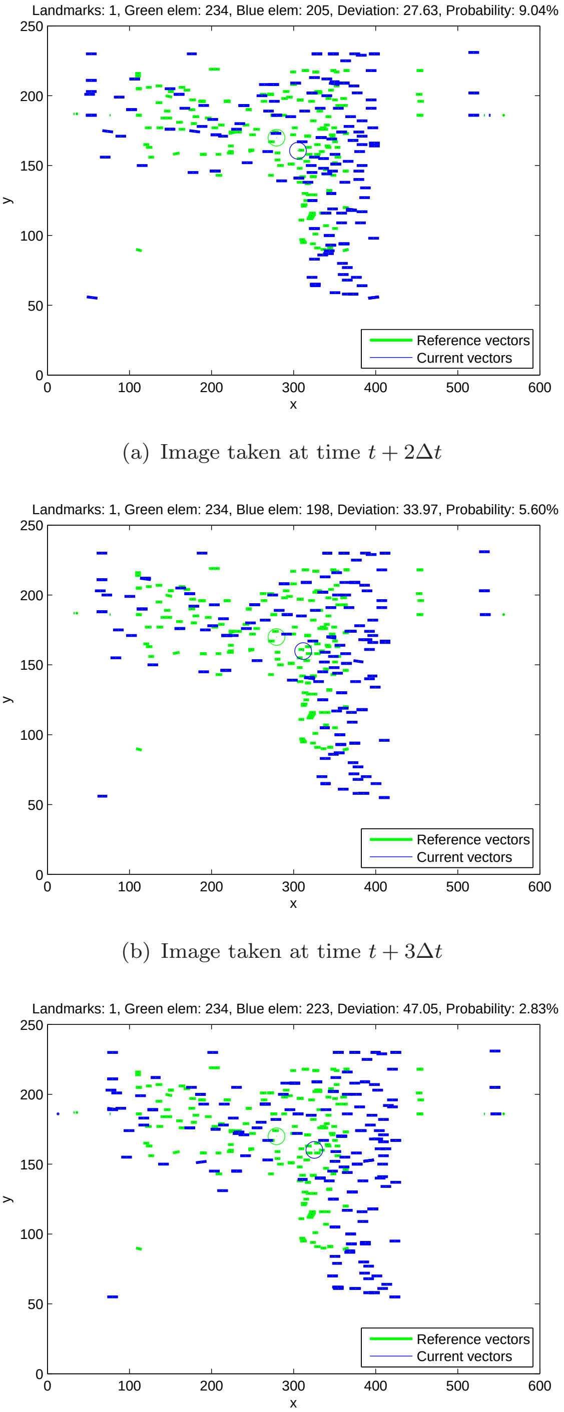


(a) Image taken at time $t$

(b) Image taken at time $t + \Delta t$

(c) Image taken at time $t - \Delta t$

FIGURE 5.20: Snapshot of reference landmark taken at a distance of 8m and a velocity of 5km/h. Reference landmark of Fig. 5.14(c) is compared against the same landmark but at a distance of 8m and a velocity of 5km/h. Similarity score is high in Fig. 5.20(a) and Fig. 5.20(b) although both distance and velocity parameters have changed. The circles in the graphs depict the mean position of the corresponding optical flow vectors. (A).



(a) Image taken at time $t + 2\Delta t$

(b) Image taken at time $t + 3\Delta t$

(c) Image taken at time $t + 4\Delta t$

FIGURE 5.21: Snapshot of reference landmark taken at a distance of 8m and a velocity of 5km/h. Similarity score is lowering as the time step the current snapshot was taken increases. The circles in the graphs depict the mean position of the corresponding optical flow vectors. (B).

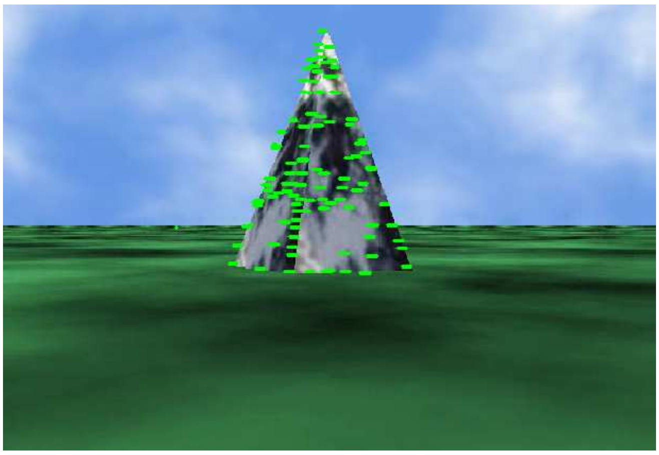
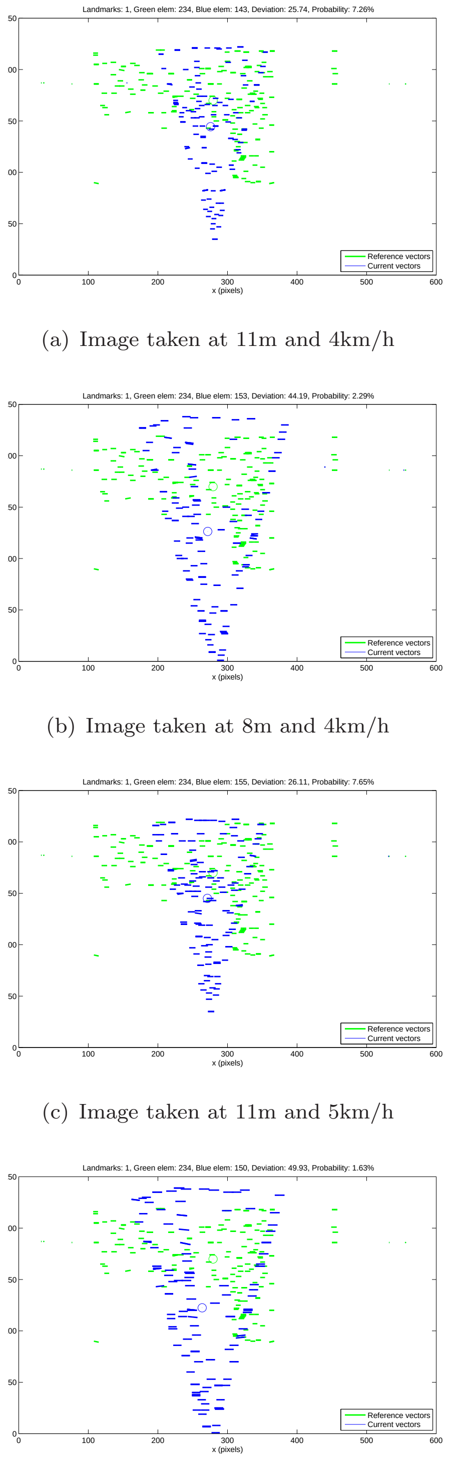


Figure 5.22: Mountain-like landmark used for comparison



Landmarks: 1, Green elem: 234, Blue elem: 143, Deviation: 25.74, Probability: 7.26%

(a) Image taken at 11m and 4km/h

Landmarks: 1, Green elem: 234, Blue elem: 153, Deviation: 44.19, Probability: 2.29%

(b) Image taken at 8m and 4km/h

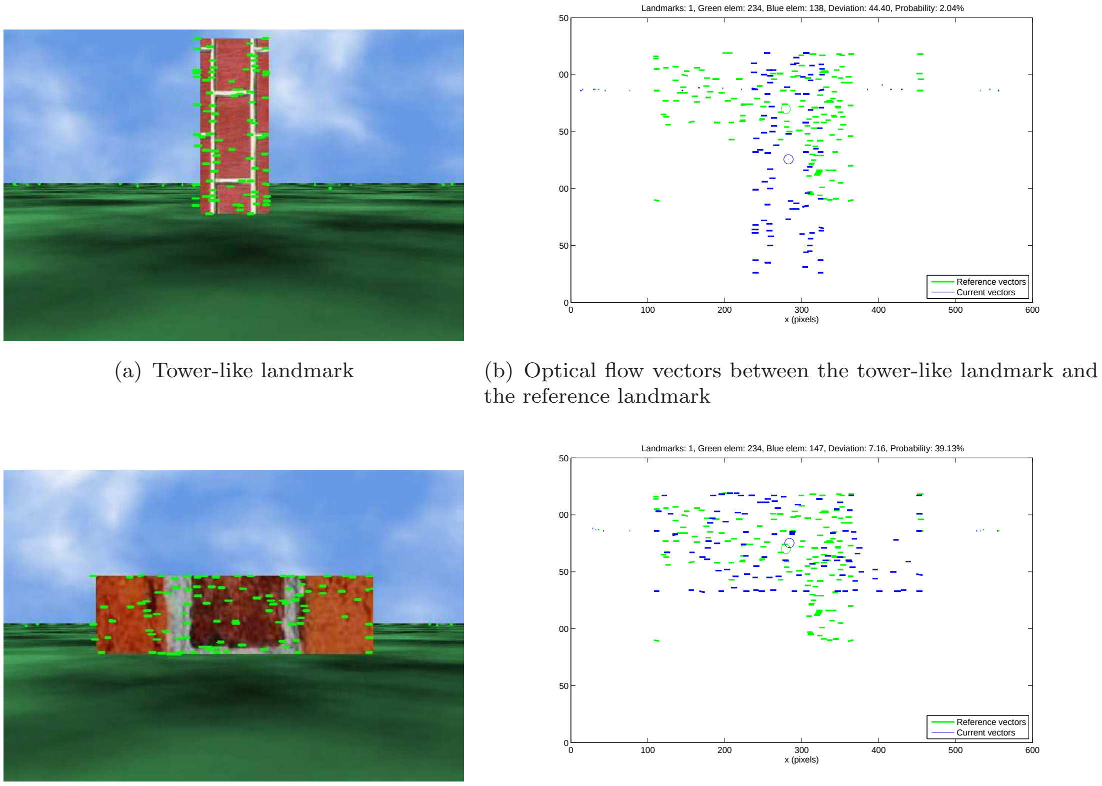
Landmarks: 1, Green elem: 234, Blue elem: 155, Deviation: 26.11, Probability: 7.65%

(c) Image taken at 11m and 5km/h

Landmarks: 1, Green elem: 234, Blue elem: 150, Deviation: 49.93, Probability: 1.63%

(d) Image taken at 8m and 5km/h

Figure 5.23: Optical flow vectors of a mountain-like landmark taken at varying distances and velocities. Current landmark is compared against reference landmark of Fig. 5.14(c). The difference between the two patterns is shown by the large deviation (distance between circles) and the large difference in the numbers of their elements. The circles in the graphs depict the mean position of the corresponding optical flow vectors.



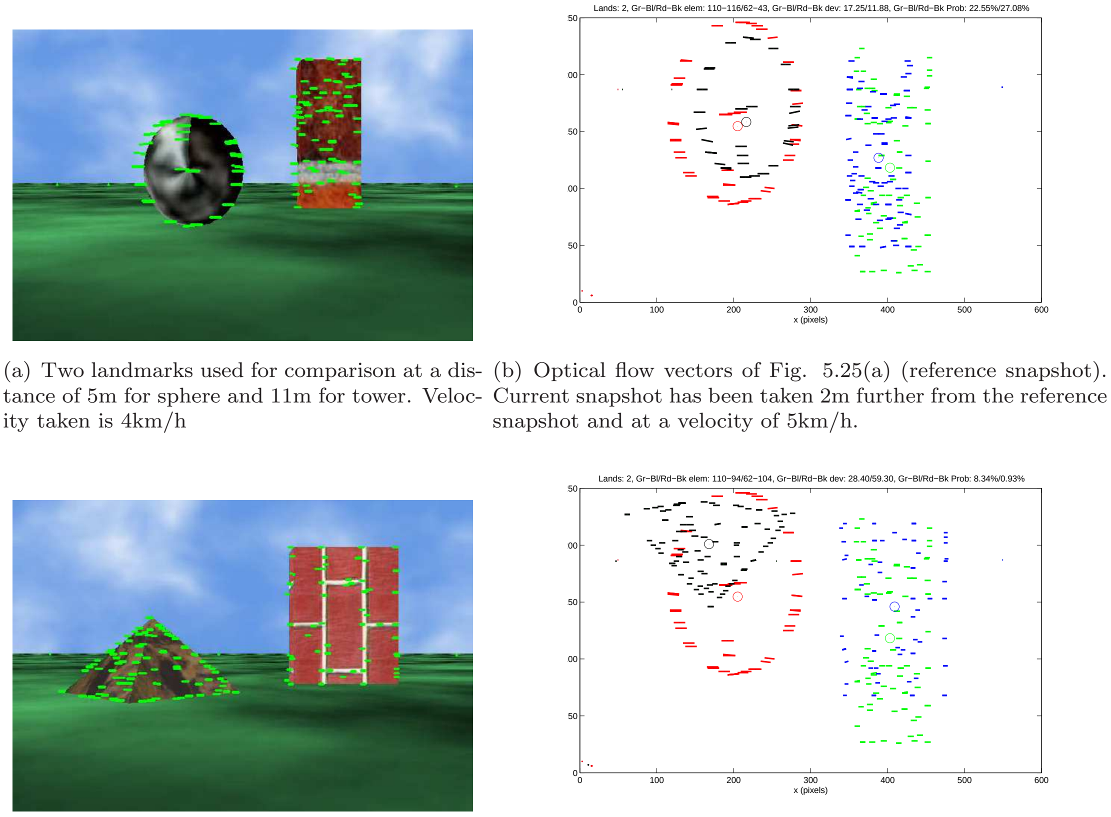
(a) Tower-like landmark

(b) Optical flow vectors between the tower-like landmark and the reference landmark

(c) Parallelogram landmark

(d) Optical flow vectors between the parallelogram landmark and the reference landmark

Figure 5.24: Comparison of optical flow vectors between two different landmarks and the reference landmark of Fig. 5.14(c). Similarity score in Fig. 5.24(b) is 2.04% revealing the difference in optical flow patterns. In Fig. 5.24(d) although the two landmarks are different the optical flow pattern are similar resulting in a high probabilistic score of 39.13%. The circles in the graphs depict the mean position of the corresponding optical flow vectors.



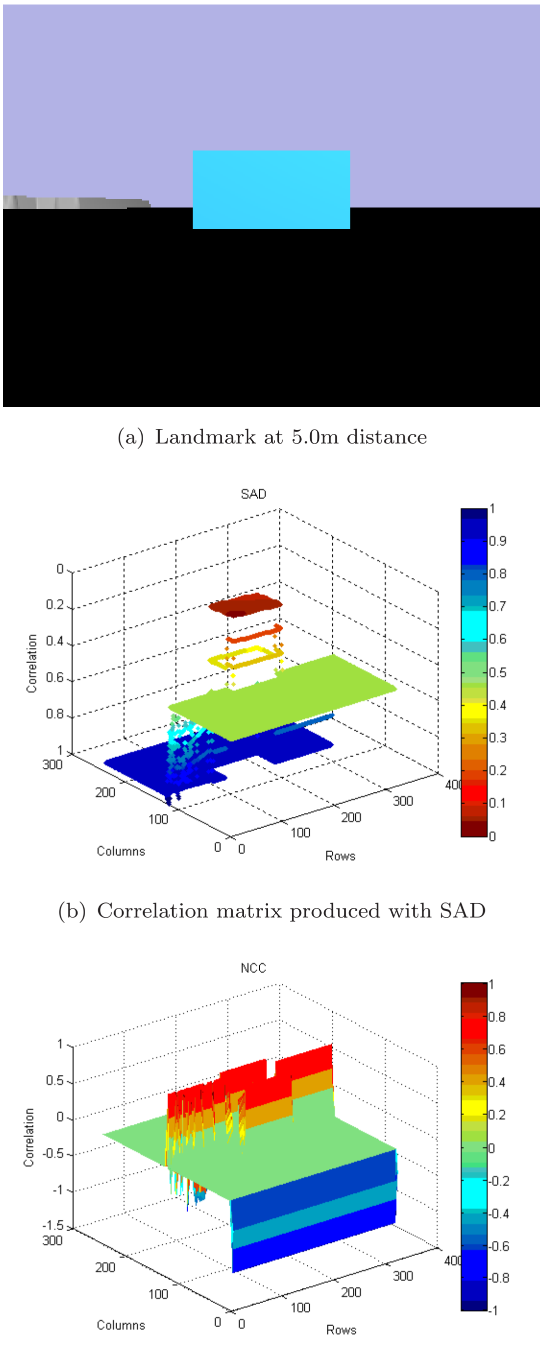
(a) Two landmarks used for comparison at a distance of 5m for sphere and 11m for tower. Velocity taken is 4km/h

(b) Optical flow vectors of Fig. 5.25(a) (reference snapshot). Current snapshot has been taken 2m further from the reference snapshot and at a velocity of 5km/h.

(c) Two different landmarks which are compared against the reference landmarks

(d) Optical flow vectors between the landmarks of Fig. 5.25(a) and the landmarks of Fig. 5.25(c) Distance from mountain landmark is 8m and for tower is 11m. The velocity the image was taken is 3km/h

Figure 5.25: Comparison of optic flow vectors between different landmarks. The circles in the graphs depict the mean position of the corresponding optical flow vectors.

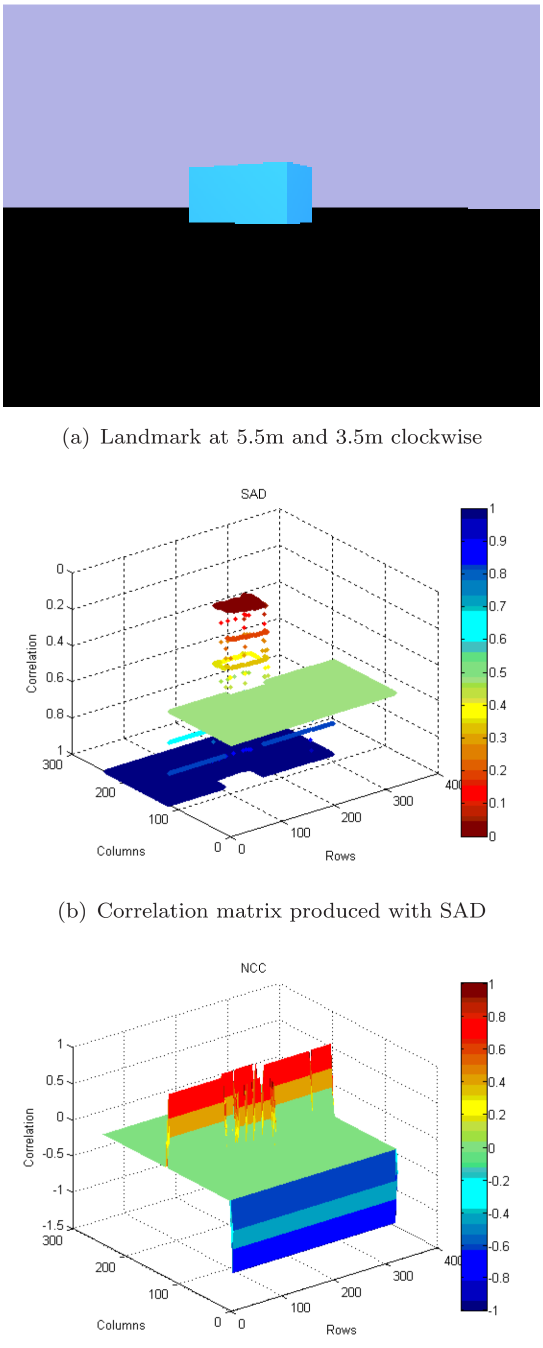


(a) Landmark at 5.0m distance

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

Figure 5.26: SAD and NCC correlation matrices of landmark image at 5.0m distance (red colour depicts absolute matching in both graphs).

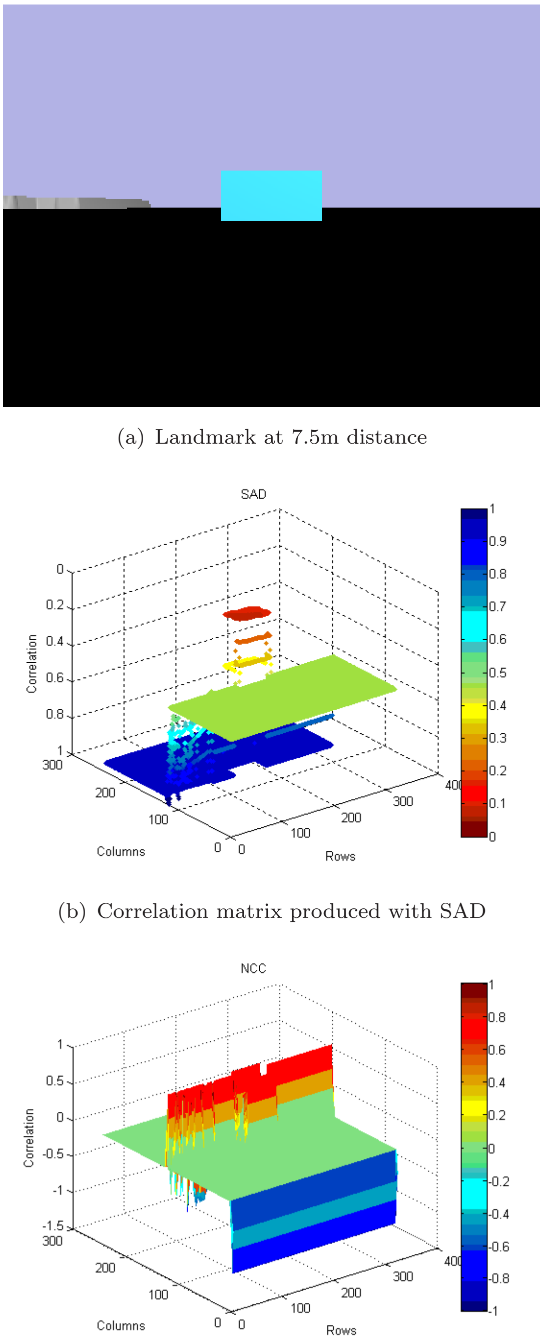


(a) Landmark at 5.5m and 3.5m clockwise

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.27: SAD and NCC correlation matrices of landmark image at 5.5m distance and 3.5m right from the centre (red colour depicts absolute matching in both graphs).

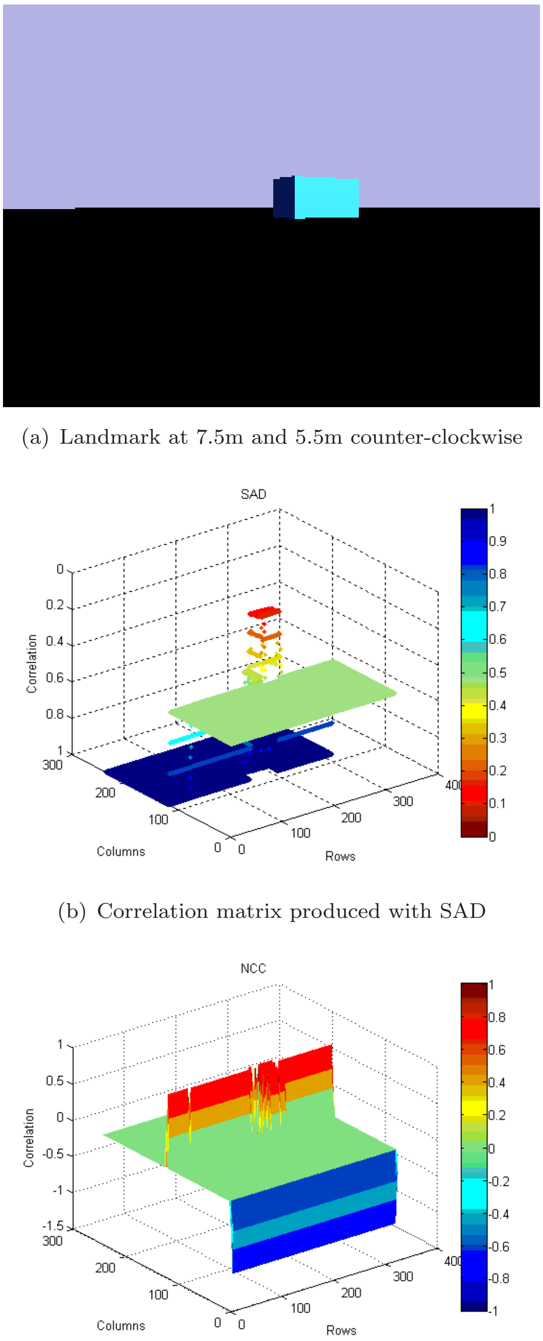


(a) Landmark at 7.5m distance

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.28: SAD and NCC correlation matrices of landmark image at 7.5m distance (red colour depicts absolute matching in both graphs).

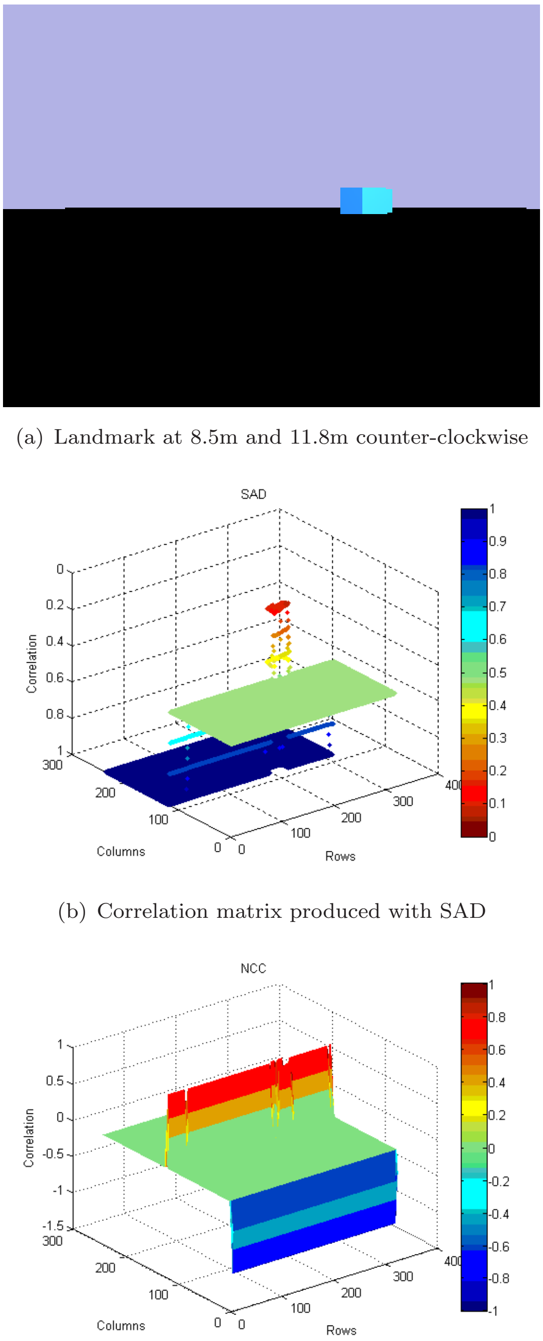


(a) Landmark at 7.5m and 5.5m counter-clockwise

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.29: SAD and NCC correlation matrices of landmark image at 7.5m distance and 5.5m left from the centre (red colour depicts absolute matching in both graphs).

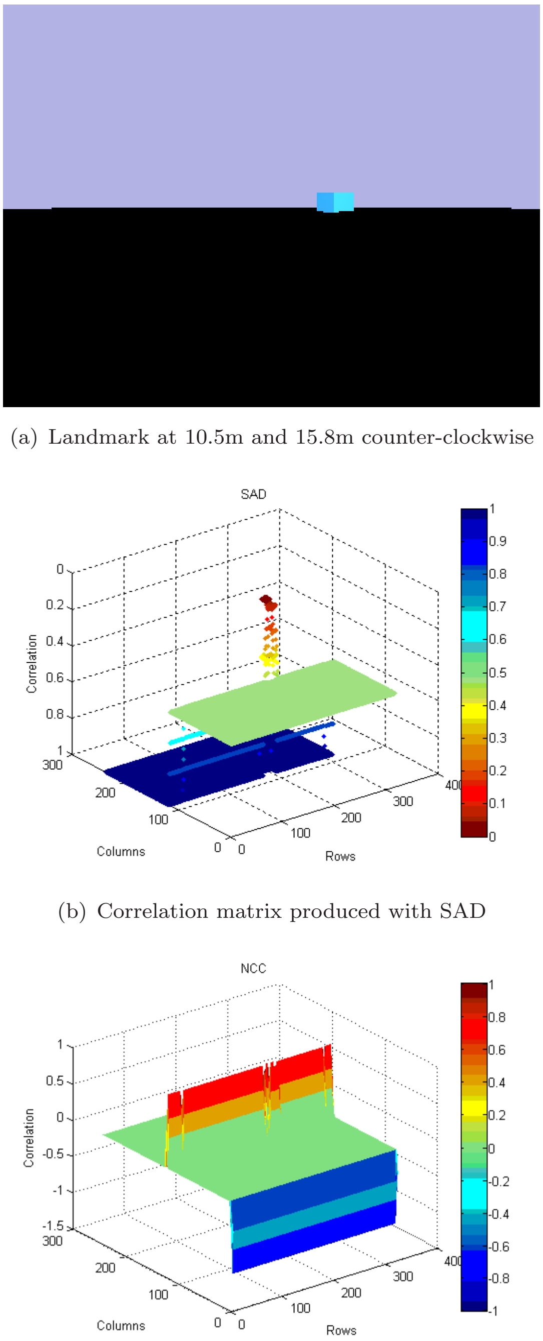


(a) Landmark at 8.5m and 11.8m counter-clockwise

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

Figure 5.30: SAD and NCC correlation matrices of landmark image at 8.5m distance and 11.8m left from the centre (red colour depicts absolute matching in both graphs).

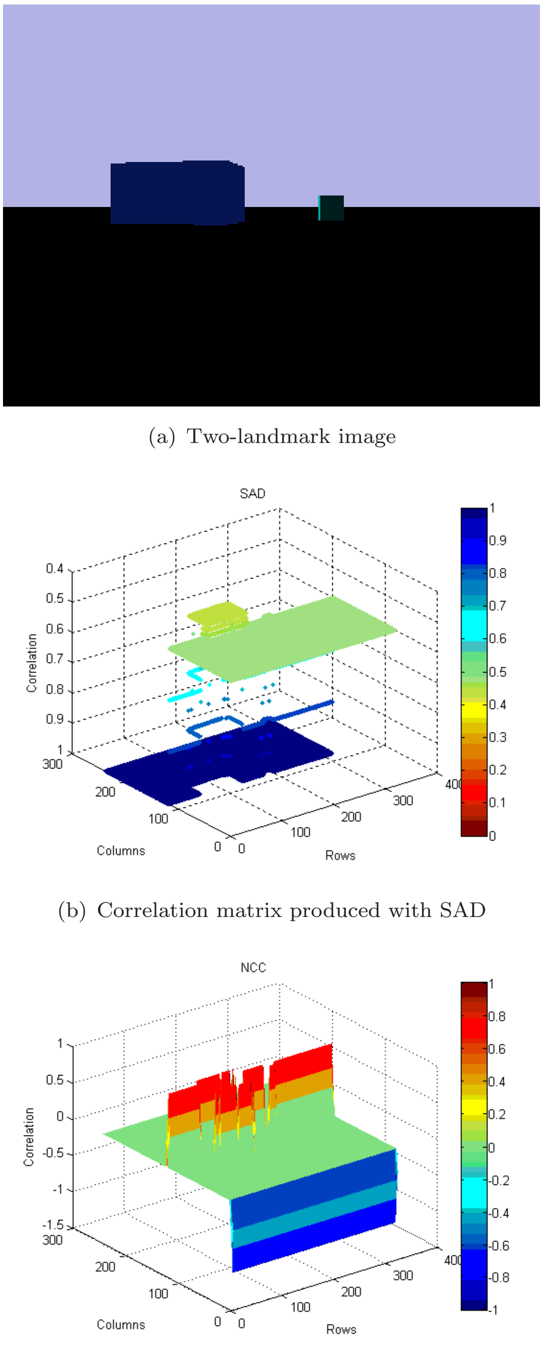


(a) Landmark at 10.5m and 15.8m counter-clockwise

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.31: SAD and NCC correlation matrices of landmark image at 10.5m distance and 15.8m left from the centre (red colour depicts absolute matching in both graphs).

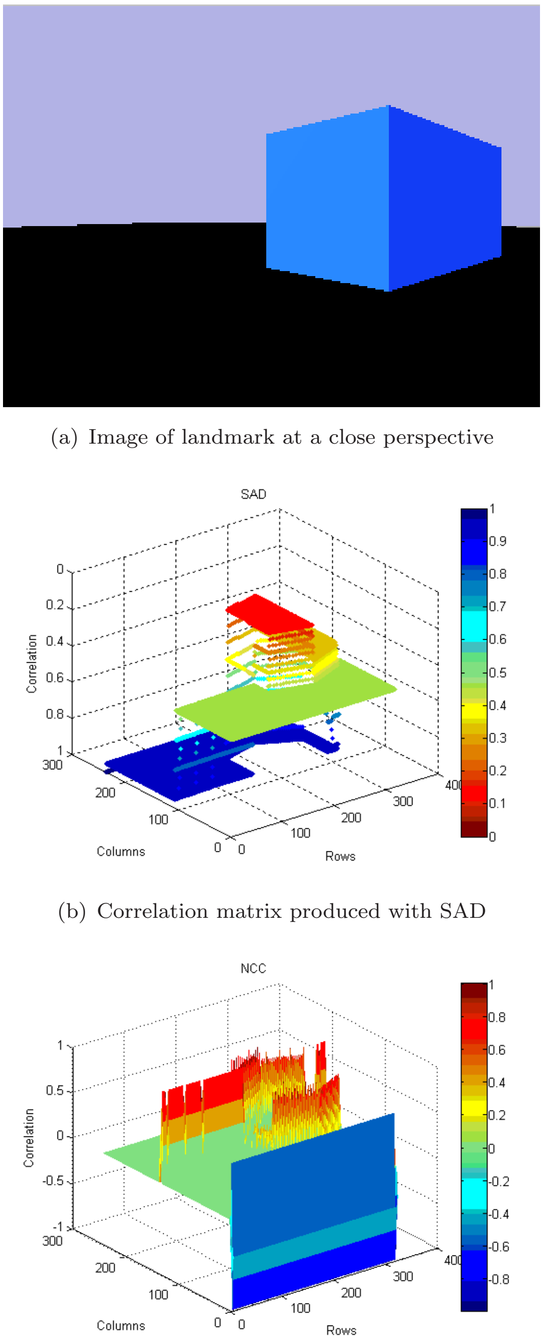


(a) Two-landmark image

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.32: SAD and NCC correlation matrices of a two-landmark image viewed. The landmarks are shadowy. The source of light is in front of the robot camera and the landmarks (red colour depicts absolute matching in both graphs).

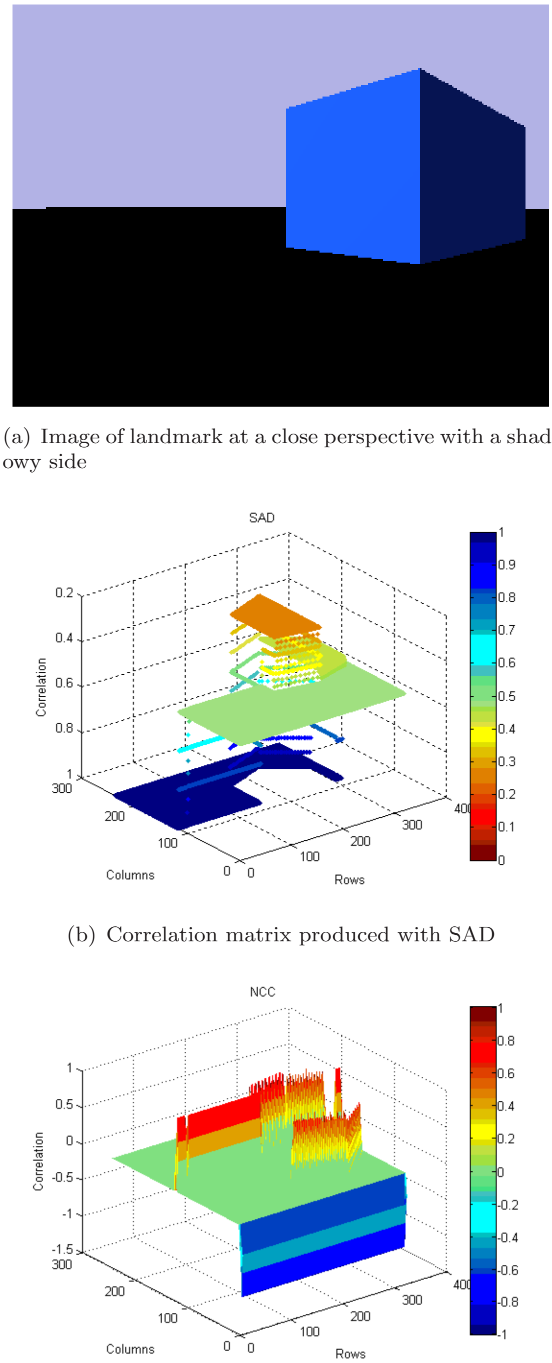


(a) Image of landmark at a close perspective

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.33: SAD and NCC correlation matrices of the landmark taken at a close angular perspective (red colour depicts absolute matching in both graphs).

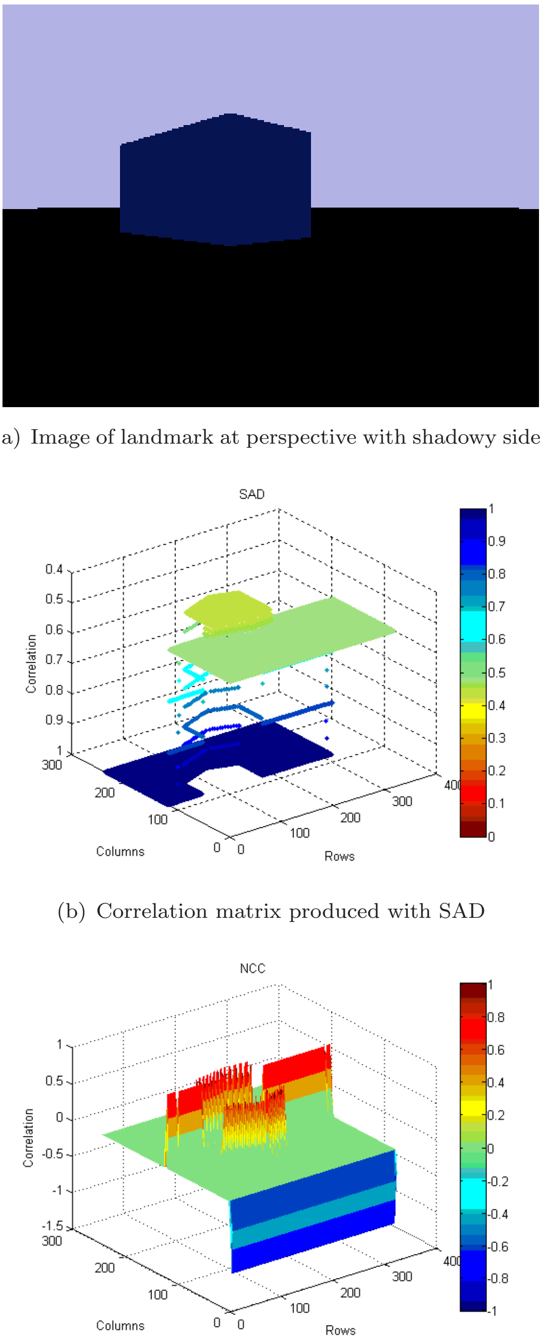


(a) Image of landmark at a close perspective with a shadowy side

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.34: SAD and NCC correlation matrices of the landmark taken at a close angular perspective with a shadowy side (red colour depicts absolute matching in both graphs).

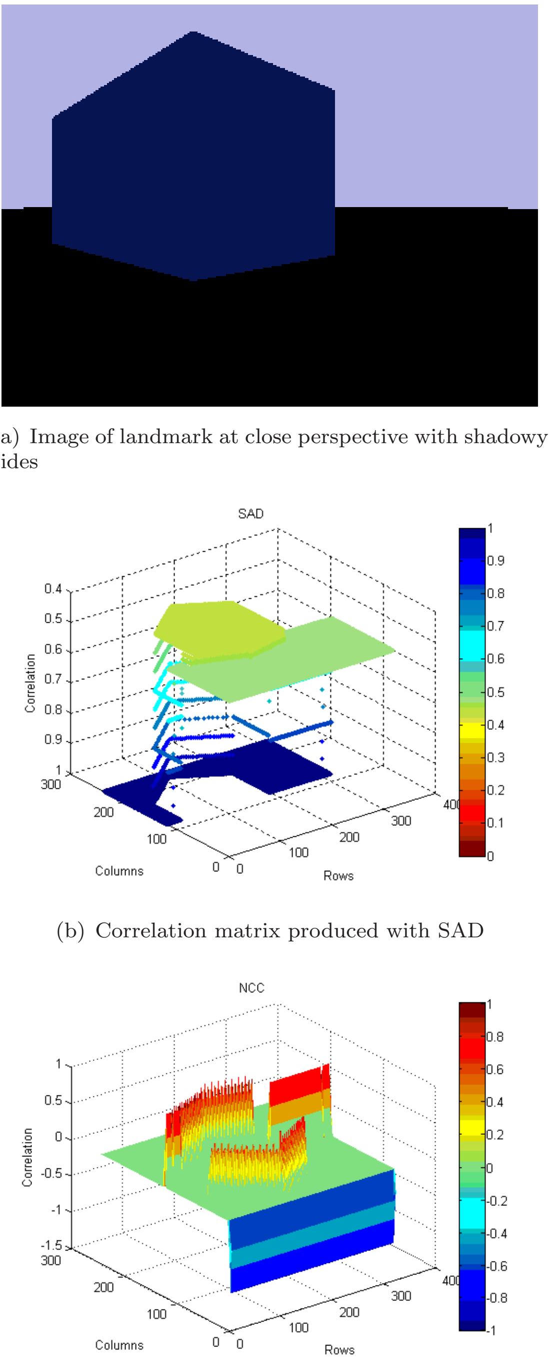


(a) Image of landmark at perspective with shadowy sides

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

Figure 5.35: SAD and NCC correlation matrices of the landmark taken at perspective with shadowy sides (red colour depicts absolute matching in both graphs).



(a) Image of landmark at close perspective with shadowy sides

(b) Correlation matrix produced with SAD

(c) Correlation matrix produced with NCC

FIGURE 5.36: SAD and NCC correlation matrices of the landmark taken at a close perspective with shadowy sides (red colour depicts absolute matching in both graphs).

# Chapter 6

# Conclusions and Further Work

## 6.1 Background

Robot homing is the process by which a robot returns to its home position. Thus far, various models have been developed to tackle the problem of homing. The most prominent ones are the *Snapshot* model and the *Average Landmark Vector* (ALV) model. Both models have been inspired by the way insects navigate. Their main advantage is the simplicity of their approach and the low computational requirements needed to perform homing. For the snapshot model, images need to be processed, matched, and stored whereas in the ALV matching occures in terms of vector components. ALV is, thus, more parsimonious than the snapshot model. Apart from the use of visual modality, both models, however, require the use of a compass. Their main limitation is that all landmarks need to be present, that is both in the current snapshot and the target snapshot. Because of this, the models can only be applied towards the end of the homing process, when the robot is close to its home position and when all landmarks are visible. On the other hand there are techniques such as Simultaneous Localisation and Mapping (SLAM) where a number of sensors are employed such as sonar, visual, laser, and odometry. The advantage of these techniques is accuracy, but computational cost is a core issue. This is the reason that faster variants of SLAM have been developed such as *FastSLAM*.

The research presented in this thesis has contributed to the state of the art by employing biological models and models which are related to the localisation and mapping methodology. The reason for using biological techniques is that provide simple, yet effective methods for navigation of mobile robots. Research on ants and bees has given an insight on how these insects carry out complex tasks such as foraging and homing. The neural circuitry these creatures carry is little but the methods they employ, effective and robust. In this thesis a number of biologically inspired methods have been deployed such as retinal landmark simulation, large-scale detection, counting of landmarks, and localisation based on the optical flow of the landmarks. The purpose was to build topological maps that will enable a mobile robot to localise with respect to its environment. The gain from such an attempt is twofold. By examining the strategies animals employ we can build better and more efficient robots, and by building such robots we can understand better how the underlying mechanisms of animals work. Further to this, a model which is related to the simultaneous localisation and mapping has been presented. In this model, metric map are built by employing a laser range finder. This model has been extended by utilising a visual sensor to track landmarks and infer depth estimates between robot and landmarks. The use of metric maps acts upon the topological maps by providing metric information among landmarks.

## 6.2 Conclusions

The literature review of this thesis has provided the state of the art in navigation of mobile robots as well as animals and insects. It has provided the basis for understanding the concepts and strategies of biological visual navigation. It has also included concepts and techniques from the Simultaneous Localisation and Mapping field. Based on that, the research question of how effectively a mobile robot can return to its home has been set. The main objectives of this research were to develop algorithms to perform navigation efficiently based on studies in biology. Thus, computational cost was the main objective of this research. For the development of the algorithms, *Player/Stage*, and partly *breve*, simulators were used. However, there exists a number of things that have to be made in order to describe as accurately as possible a virtual environment. In general, the structure of the environment was kept simple. Nevertheless, noise was modelled in order to make more realistic the environments. Furthermore, the virtual environment was kept simple for the purpose of understanding the navigational mechanisms that underlie the biological organisms. This is one of the main novelties of this work. Apart from designing an efficient robotic system for performing homing, the results from the methods followed may reveal the strategies that insects perform in order to navigate. Such a strategy is the optical flow patterns generated by the motion of the virtual or physical agent and their use for localisation.

In Chapter 4 it was presented the localisation and mapping model based on a laser range finder. The novelty of this model lies in the fact that a single sensor, namely a laser range scanner can provide the localisation of the robot and mapping of the environment. The algorithm selects the nearest landmark to the robot and tracks it until laser's side rays detect the landmark. The advantage of the approach is that it does not use a large number of resources unlike many algorithms in this field. On the other hand, the limitations of the approach are that it is environment-dependent, that is, an environment with geometrical entities is suitable for this type of model. Furthermore, it was shown that optimal paths can be chosen through the navigation of the robot. This advantage holds so long as the laser rays are not occluded by the expansion of the obstacles. In addition to this, a visual sensor has been added to the model with the purpose of providing depth estimates between the robot and the landmark so as the robot can navigate through environments where landmarks are not always distinguishable. In particular, two algorithms were developed for this purpose, a least squares and an optical flow. The combination of both methods has proven to be quite effective in estimating depth. This model provides the metric maps to the robot.

In Chapter 5 a novel representation of the environment has been used. Optical flow patterns act as maps for the navigation of the robot. During the outbound trip of the robot, a 'fingerprint' of the environment is created based on the optical flow patterns of the landmarks. Based on a training algorithm the model infers a similarity score between the patterns of the current snapshot (the snapshot taken during the homing process) and the reference one. One advantage of the method is that it does not make use of any other information apart from the optical flow. Moreover, images are not stored as most of traditional methods do. Only the properties of the flow vectors are stored, that is their mean position and their number, and any comparison is done on the basis of vector properties. This model proves to be parsimonious not only because of its low memory requirements but also because it does not make use of any other sensor apart from a visual one. This model provides the topological maps of the environment. At the end of the homing process the robot matches the snapshots of the TBL approach with its current view in order to infer home position.

The simulations of this model have shown that it is possible honeybees to use optical flow patterns to localise. So far most of the biologically-inspired approaches could benefit only one way from the application of biology onto robotic systems. Robust and efficient biologically-inspired robots have been designed and built but very little understanding of the underlying biological mechanisms has been gained. This research gives a new insight on how insects, and in particular honeybees navigate. Furthermore, this biological model does not consider the properties of the landmarks, that is colour, shape, and size, but the properties of the camera motion. This property is essential for the honeybees since it is known that they use optical flow at least for obstacle avoidance. Yet, Avargues-Weber et al. (2009) reveal that honeybees are capable of discriminating faces. It could well be the case of optical flow pattern recognition. Kendoul et al. (2009) write in the future work of their paper that an extension of his work would be to investigate the mapping of the environment using optical flow techniques.

Altogether, both the optical flow model and the model based on the laser range finder allow us to make significant performance gains over existing approaches. The objectives which were set and discussed at the first chapter have been met. The next section presents some ideas which can extend further the current models.

## 6.3 Further Work

The aim of this work was to develop algorithms that will enable a robotic system to perform efficiently the homing task. The following are some of the ideas that can be implemented and extend the scope as well as the capabilities of the model.

- Develop a Bayesian approach to localise the robot based on the optical flow maps. In this case a prior distribution needs to be calculated and a likelihood to be determined. The posterior distribution will enable the model to adapt and improve as new observations enter the model. This, will therefore, expand the navigation abilities of the robot to various types of environments.
- Perform SLAM techniques for localisation of the robot. This techniques, however, will differentiate the scope of the model since core SLAM techniques involve acquisition of data from sensors such as dead reckoning and sonar. This will increase the complexity of the model as well as its computational cost, but it is expected to perform well in complex and dynamic environments.
- Integrate colour patterns onto the optical flow model. This would be an interesting aspect to investigate since it is known that honeybees have acute vision on the main colours.
- Develop a dense tracking algorithm and compare it with the current sparse tracking algorithm. As has been mentioned in earlier chapters a dense tracking algorithm may perform better but at the same time this would increase the computational complexity of the whole model.
- Develop a better clustering algorithm for landmark counting. For the purpose of this work where the number of landmarks is not known in advance nor which vectors belong to which landmarks, clustering becomes a difficult and at the same time challenging problem. An unsupervised clustering methods needs, therefore, to be developed.
- Apply the optical flow model on a multi-agent system in order to infer the exact position of a landmark.
- Perform spiral techniques to search for lost landmarks. In Chapter 2 the spiral patterns ants perform when lost were discussed. It can be shown using simulation what type of spiral patterns ants perform more closely and how effective this is compared to other search techniques. Some known spirals are the *Archimedean spiral*, *Fermat's spiral*, *hyperbolic spiral*, and the *golden spiral* which is a special case of the *logarithmic spiral*. *Golden spiral* is found in nature and in things like sunflower heads, nautilus shells, and others.

# Appendix A

# Localisation and Mapping Using a Laser Range Finder: A Goal-Seeking Approach

Sotirios Ch. Diamantas and Richard M. Crowder. ICAS 2009, Valencia, Spain, 2009.

**Abstract**

*In this paper we examine the problem of localisation and mapping of an unknown environment using data from a laser range finder. In order to support our method we detect landmarks in the environment using the same laser finder. For the localisation and mapping process to take place we assume that the mobile robot will follow a path until a landmark is observed by the laser scanner. Our approach alleviates the requirement to provide odometry or other information. In addition, an efficient path is sought to reach target location. An inherent property of this is obstacle avoidance. The simulated experiments presented in the paper validate the effectiveness of our approach.*

## A.1 Introduction

For a mobile robot to build a map of an unknown environment an accurate estimation of the position of the robot is required as well as a means for effectively mapping the environment. Localisation and mapping can be considered to be a "a chicken and egg" problem, requiring both accurate position estimates of the robot and of the surroundings. A laser range finder is highly suitable for that purpose due to its strength in estimating distances accurately while unburdening the system from problems arising in other sensors like infrared or sonar. In fact, there is a large number of works that utilise laser range sensors for obstacle avoidance Ulrich and Borenstein (1998); An and Wang (2004); Koren and Borenstein (1991); Martinez et al. (1998).

In this work no prior information about the environment is assumed. The only hypothesis made is that the robot knows where it should head to. Thus, we feedback the robot with a goal that has to be sought. However, even this hypothesis can be ruled out so long as the robot navigates without any restrictions, i.e., without having to pursue an effective path. In such a case the robot can navigate towards places where there are distinguishable points by the laser finder, for example corners, and drive among such 'places'.

In this paper we present a navigation method based on the idea of Vector Field Histogram (VFH) Ulrich and Borenstein (1998). The robot scans the environment using the laser sensor and based on the measurements taken an efficient path is sought. In order to infer an efficient path the target point is assumed to be known. We can, therefore, make a supposition that the goal position is known by its $x, y, \phi$ coordinates or that the goal position can be seen through a vision sensor and an estimate of its distance or direction can be taken. Next, the robot will try to identify 'distinctive' landmarks in the scanned environment; these are mainly corners, and can be identified by the range of the neighbouring ray values. The robot will thus extract the shortest obstacle-free path to the target defined by the rays of laser scanner. While traversing the chosen route, the landmark is tracked down by the laser scanner's rays until it is 'met' by the side rays of it. We use the side rays of the laser scanner to identify the landmark as they form a 90° angle with the chosen path. The sides of the right triangle formed by the side rays of the scanner, the landmark, and the position of the robot at which the landmark was first detected are computed with the trigonometric functions.

The process of finding a 'distinctive landmark' is critical for the path to be extracted as is needed to be as close to the autonomous agent as possible. Fig. A.1 demonstrates this process where the robot at time $t_0$ scans the environment and selects the nearest landmark to it, depicted by the bold scan line. At time $t_1$ the robot is scanning again the environment using the laser scanner and an updated second path is chosen to reach the target at point denoted by the 'X' symbol. The robot could have missed to reach the target and end up in a blind alley had it not selected a nearby landmark.

The primary purpose of this work is to tackle one of the core problems in robotics science, namely localisation and mapping using minimal sensing. It is not uncommon that a critical sensor to the system may fail and thus having to bypass it using alternative sensors and methods. Such cases become of high importance if the autonomous agent acts in remote or hostile environments. Moreover, sensors like GPS cannot operate in environments like, indoors, underwater, or the outer space. Therefore, we have approached the localisation and mapping problem using only a laser scanner sensor to infer accurate maps of the environment while at the same time localising the robot.

This paper is comprised of five sections. Following is section 2 where related work is presented. Then, in section 3 we present and analyse the methods that have been implemented to localise the robot and map the unknown environment. In section 4 we present and discuss the experimental results. Finally, section 5 epitomizes the paper with a brief discussion on the conclusions derived from this work and a future work.

## A.2 Related work

Localisation and mapping has been at the forefront of robotics research the last decade. The work presented in this paper is closely related to the Simultaneous Localisation and Mapping (SLAM) Smith and Cheeseman (1986) problem in which a robot has to build map while at the same time estimating its position relative to the map. Most of the approaches make use of Kalman Filter (KF) and Extended Kalman Filter (EKF) Dissanayake et al. (2001); Leonard et al. (1992), Armesto and Tornero (2004); Choi et al. (2007). Under these methods a matrix representing the robot and landmarks' position is established. As the number of landmarks increases so does the matrix resulting in a computational expensive solution. An alternative solution to KF and EKF is the use of particle filters Fox et al. (2001) and Monte Carlo methods Thrun et al. (2000, 2001).

FastSLAM Hahnel et al. (2003); Montemerlo et al. (2002) is another method that integrates particle filters and Extended Kalman Filters. FastSLAM tries to alleviate from the problem of *data association*, that is, the problem where landmarks look alike which is prevalent in the previous methods. Most of the above mentioned methods use mainly odometry information and laser sensors to support the SLAM methodology. Nonetheless, there has also been a SLAM-based method that employs a single camera Davison et al. (2007) to infer the 3D trajectory of a monocular camera in an unknown environment. This vision-based SLAM is widely known for years as the 'structure from motion' problem and it has been researched in parallel and in ignorance of the robotics community. Other sensors which have been used for localisation include Radio Frequency Identification (RFID) technology. For example Hahnel et al. (2004), proposes a model based on RFID technology and a laser-based FastSLAM approach to effectively determine the location of RFID tags.

In Amigoni et al. (2004) the authors proposed a method for map building without using any odometry information. Their method builds a geometrical global map based on various scans of the environment taken at different instants of time. They try to integrate the maps using three different methods, namely sequential, tree, and pivot methods. However, they do not keep track of the position of the manually driven robot. An approach which uses Dynamic Programming (DP) for the real-time self-localisation of a robot is explained in Einsele (1997). In this work a panorama laser range finder (PLRF) is employed. Similarly to the previous work, a matching between present preprocessed scans and already stored scan data is taking place. The task of preprocessing method is to extract line segments from the acquired range data. Moreover, in their approach

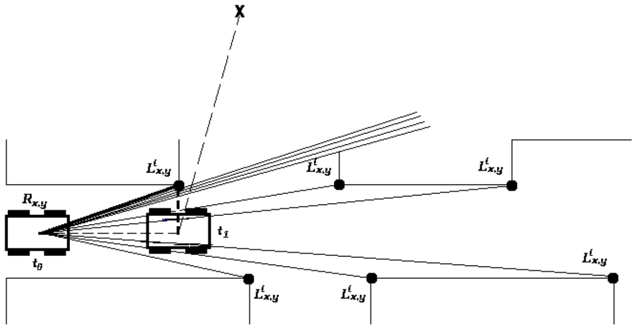


Figure A.1: Robotic agent having selected the nearest landmark which leads to an efficient path to the target point 'X'.

they make use of local coordinate systems linked together by topological information. No global environmental map is built, but they rather focus on local 'distinctive' places to build local maps for self-localisation.

## A.3 Methodology

In this section we describe the methods we have followed to tackle the problem of localisation and mapping. No a-priori knowledge of the environment is provided. However, as already stated we adopt a goal-seeking approach and a competent path is selected which is defined by the laser scanner rays. The map of the environment is initially empty and is built up as the navigation of the autonomous agent proceeds. Fig. A.2 depicts the structure of the localisation and mapping algorithm.

During the first phase the robot collects raw data from the laser range finder, and in the next one the obstacles of the environment are expanded, see Fig. A.3, according to the radius, $r$, of the robot using the equation of the circle (A.1),

$$(x-h)^2+(y-k)^2=r^2, \tag{A.1}$$

where $h$, $k$ is the centre of the circle $C$, in this case the point at which the ray of the laser hits on the obstacle, and the equation of the line (A.2),

$$y=mx+b, \tag{A.2}$$

that represents a laser ray. We compute the slope (A.3) of the line, i.e., ray,

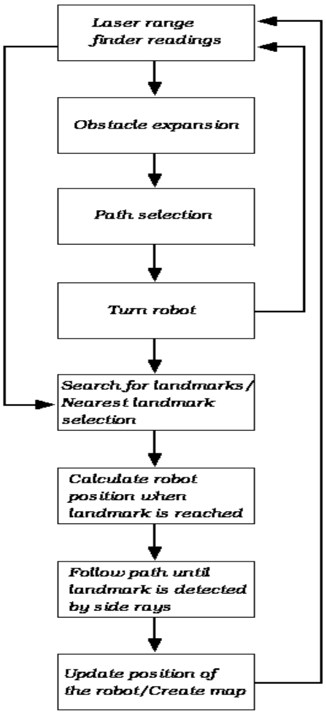

Laser range finder readings
Obstacle expansion
Path selection
Turn robot
Search for landmarks/ Nearest landmark selection
Calculate robot position when landmark is reached
Follow path until landmark is detected by side rays
Update position of the robot/Create map

FIGURE A.2: Flowchart of the various stages of the localisation and mapping algorithm.

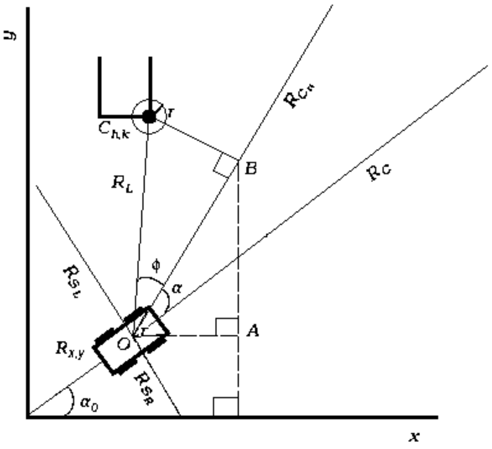


FIGURE A.3: Robotic agent having identified a new path, $R_{C_n}$, and a landmark, $C_{h,k}$. This figure shows how the localisation coordinates are computed from the triangles $OBC$ and $OAB$.

$$m = \frac{C_k - R_y}{C_h - R_x}, \tag{A.3}$$

by knowing the current position of the robot $R_{x,y}$ and the end position of the ray, i.e., $C_{h,k}$. Next, we substitute equation (A.2) into the second part of (A.1) having (A.4),

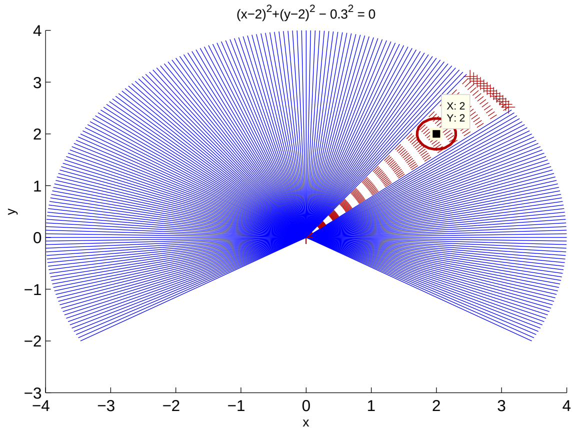


Figure A.4: A simulated scan sample with a detectable area of 240° and a detectable distance of 4.0 metres. An expanded point by 0.3 metres is shown with the laser rays falling into the circle being marked.

$$(x - h)^2 + (mx + b - k)^2 = r^2. \tag{A.4}$$

Thus, from (A.4) we end up having a quadratic equation (A.5),

$$ax^2 + bx + c = 0, \tag{A.5}$$

which we solve it in order to obtain the points in which the rays intersect (or hit) the obstacles expanded by the radius $r$. Thus, if discriminant, $\Delta > 0$, then the ray intersects the expanded obstacle in two points, whereas if $\Delta = 0$, there exists a tangent ray to the circle. If $\Delta < 0$ then the ray is not intersecting any point of the circle formed by the expansion of the obstacle. Fig. A.4 shows the rays of the laser scanner one of which hits at an obstacle with coordinates $x = 2$, $y = 2$. The circle represents the expansion of the obstacle by the radius, $r$, of the robot, in this case 0.3 metres. The marked lines representing the laser rays that fall within the expanded obstacle do not thus provide a safe path for the robot. All other rays, hence paths, would provide a safe path if they were to be taken. Fig. A.5 shows the outcome of applying the above algorithm to the laser data with added noise.

Upon expansion of obstacles the path selection module takes over, Fig. A.2, and an efficient path is chosen based on the proximity of the laser ray to the target point. As written earlier, the coordinates of the target are known to the system. This however, could have been omitted had we used a compass or a vision sensor to know at any instant of time the direction to the target. So long as the path is selected, the robot rotates around its axis until its central ray, $R_C$, Fig. A.3, points to the selected path, $R_{C_n}$. The next step involves updating the laser data by taking a new laser scan after the robot has rotated by angle $\alpha$. At this point the landmark search module is taking over which identifies corners, jumps, and discontinuities in the laser scan. Such points

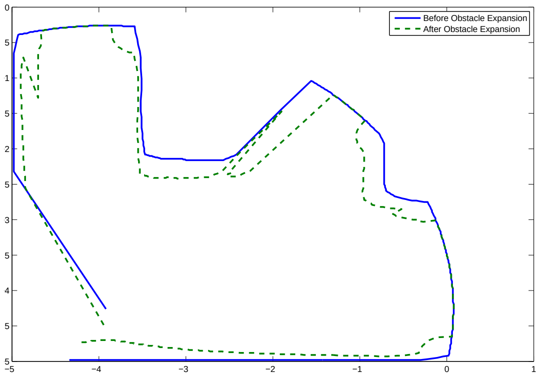


Figure A.5: Expansion of obstacles.

act as ‘landmarks’ and are identified when a ray being equal to the maximum range, say set $A$, and its neighbouring $n$ rays do not belong to the set $A$, i.e., they hit on an obstacle. Therefore, the mathematical formulation of this is, $ray_x \in A$, and $ray_{x+n} \notin A$ or $ray_{x-n} \notin A$ constitute a ‘landmark’.

Furthermore, the landmark search module will eventually select a single landmark at each time step for navigation. The selection of this landmark is based upon its proximity to robot location. This is happening because a shorter path to the target may be discovered after the approximation of the agent to the landmark. Therefore, we use the Euclidean distance, $d$, between the landmark location, $(L^i_x, L^i_y)$, and the robot, $(R_x, R_y)$ to find the nearest landmark. Fig. A.1 shows an example of landmark selection among a set of detected landmarks. The Euclidean distance is described by (A.6)

$$d = \min i \sqrt{(R_x - L^i_x)^2 + (R_y - L^i_y)^2}. \tag{A.6}$$

The area of searching for landmarks is less than 90° from the central ray, $R_C$, Fig. A.3, as the robot is using its side rays, namely, $R_{S_L}$ and $R_{S_R}$, to detect the landmark and build a map. At this point we use a local frame of reference to calculate distances to landmarks. Fig. A.3 shows how the method works. The robot after having inferred a new path it rotates by angle $\alpha$, that is, from $R_C$ to $R_{C_n}$. At this point its $R_L$ ray detects and selects landmark $C_{h,k}$. The sides of the triangle $OBC$ named $CB$ and $OB$ are calculated using the trigonometric functions $\sin\phi$ and $\cos\phi$.

Having found the coordinates of local frame of reference we add them up to the global frame of reference. Again, Fig. A.3 depicts the variables involved. This is done by adding the angle by which the agent has rotated, that is $\alpha$ degrees, to initial pose $\alpha_0$, thus inferring the pose of the robot. We then find the projection of side $OB$ to the *x-axis* and *y-axis*. The sides of the triangle $OAB$ are again calculated with the *cosine*

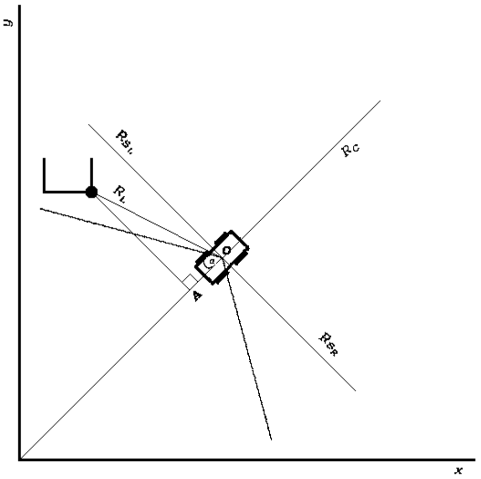


FIGURE A.6: An example of how the location of the robot can be calculated after having missed the landmark by the side ray, $R_{S_L}$. The landmark is, however, still detectable by the neighbouring rays of the laser scanner.

and *sine* trigonometric functions. Their outcome will thus give us the $x, y$ coordinates of the point $B$ which are added to the initial $R_{x,y}$ coordinates of the robot.

In order for the robot to follow the path from point $O$ to point $B$ the landmark at point $C_{h,k}$ is tracked down by the neighbouring rays of the $R_L$ ray as the robot proceeds. At point $B$, as Fig. A.3 shows, the side ray, $R_{S_L}$, will detect the landmark by the range of the $R_{S_L}$ ray which should approximately be equal to $CB$ found earlier. Due to noise, it is quite probable that the landmark be missed. For this reason we have simulated a laser scanner whose field-of-view (FOV) is 240° so if the landmark is missed by the side ray, $R_{S_L}$, it can be detected by the neighbouring rays with angle larger than 90°. Fig. A.6 shows this scenario.

At this time step the robot will localise itself with its global $x, y$ coordinates and create a new map of the environment. Having reached the final step of the algorithm, the agent initiates a new laser scan to proceed with the navigation process.

## A.4 Experimental results

The localisation and mapping algorithm has been developed on the *Player/Stage* 2D simulation environment Player/Stage (2006). The simulated devices consist of an *Erratic* mobile robot and a *HOKUYO URG04LX* laser range finder with 240° field of view, 4.0

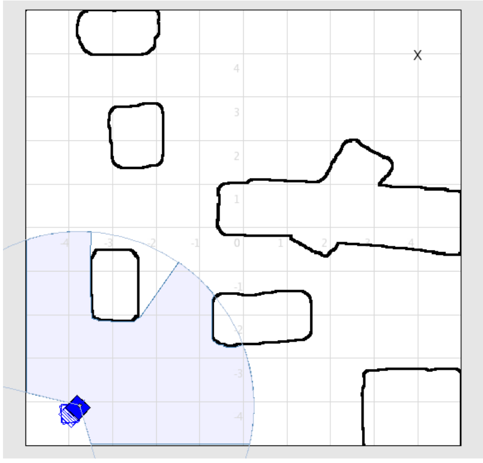


Figure A.7: Snapshot of the simulated environment.

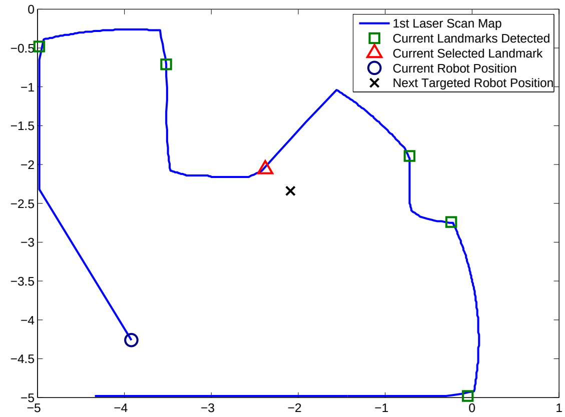


Figure A.8: First scan of the environment.

metres maximum range, and 685 samples per scan. The angular resolution was set to 0.35°. Fig. A.7 shows a snapshot of the simulated testing environment, the size of which is 10.0 metres by 10.0 metres. The robot is on the left lower corner with a delineation of the laser scanner.

Fig. A.8 shows the first scan at position $R_{x_0} = -3.92$, $R_{y_0} = -4.26$. The target position has been set at $x = 4.00$, $y = 4.00$. In this first laser scan there have been detected six landmarks of which, the nearest one, is selected for reference which is at position $x = -2.38$, $y = -2.06$. The robot will try to re-detect the landmark from position denoted by the symbol × at location $x = -2.09$, $y = -2.34$.

In Fig. A.9 the robot has successfully recognised the landmark and has updated its location coordinates; a new scan has performed with two landmarks present one of which again is selected for the navigation.

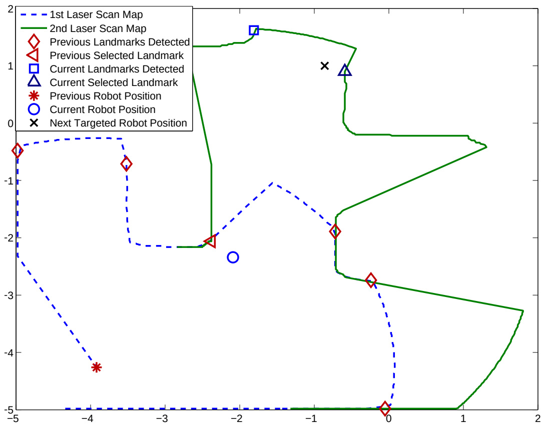


FIGURE A.9: Second scan of the environment.

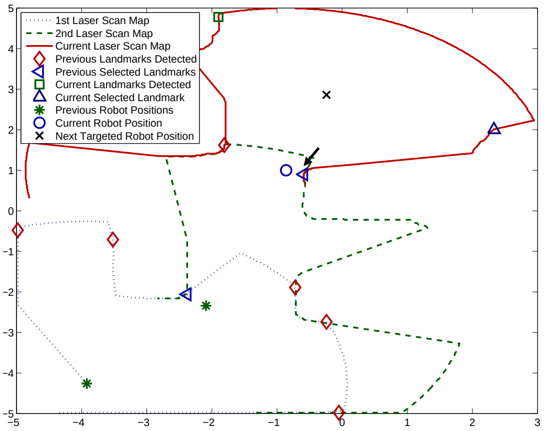


FIGURE A.10: Third scan of the environment.

In Fig. A.10 the robot has traversed the environment even more and has detected two other landmarks. At this point, although the target position is at $x = 4.00$, $y = 4.00$, the robot has selected its next targeted position, denoted by × symbol, to be further upwards instead of being on its right side. This occurs because the expansion of the obstacles obscures some laser rays that form a path in the right-hand side of the robot. The arrow shows the position, in particular the corner point, at which this occurs.

It should be noted that during the landmark selection process we use the map that does not contain the expanded obstacles. The map with the expanded obstacles is used in the path selection process. In Fig. A.11 the robot has reached its desired position and a new scan has been taken which has detected two landmarks in the environment, one of which is again selected.

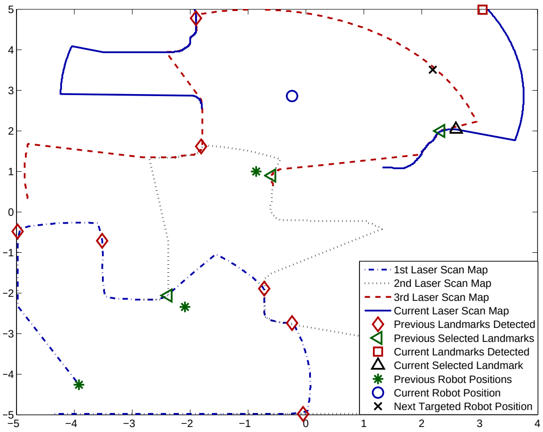


Figure A.11: Fourth scan of the environment.

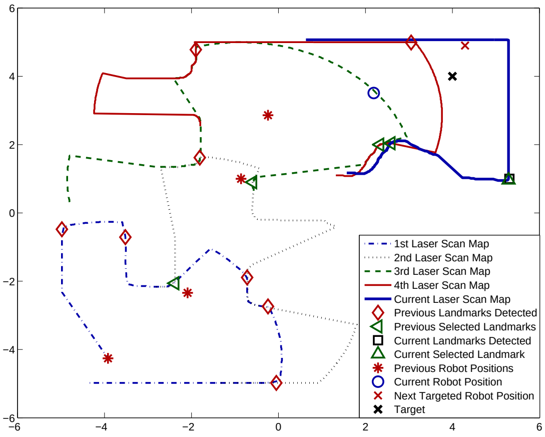


Figure A.12: Fifth scan of the environment.

In Fig. A.12 the robotic agent has almost reached its target. The boldface × denotes the target position. However, in this figure we see the influence of noise into the navigation process. There exists a map drift between the previous laser scan and the current one. This noise effect has caused the agent to select its next target point a little further away than the final target.

Fig. A.13 shows all previous laser scans and the route of the robot as calculated by its localisation system and as it appears by the GPS system. From the graph it is seen that the autonomous agent has performed quite well in the first scans. Nevertheless, there is a small drifting in the accuracy of the localisation system in the last phase of the navigation process caused mainly by the noise. In this last phase the landmarks selected, as can be seen from Fig. A.13, are not prominent corners, but they rather occur in a curvy slope.

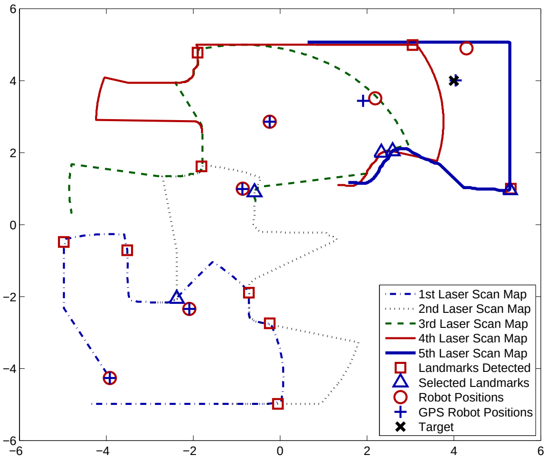


FIGURE A.13: A performance comparison between the localisation system of the robot and the GPS.

## A.5 Conclusions and future work

In this paper the problem of localisation and mapping has been addressed. The key idea behind our work is that no prior knowledge of the environment is used nor any odometry information. In fact, the odometry has been substituted by a laser range scanner. Its main purpose is to localise a robotic agent and map the environment while at the same time selecting efficient paths for driving the robotic agent to a target place. An inherent behaviour is obstacle avoidance. This is performed while path planning obstacle-free routes to desired locations.

The problem of localisation and mapping is at the heart of robot navigation and it has been approached through many ways. Our method promises to be efficient and accurate so long as the environment permits it. For example, an environment in which wide corridors occur results in erroneous estimates of the location of the robot. However, our approach is described by its simplicity and its efficiency. The simulated environment we have used to support our methodologies proves our assertions.

Future work will focus on larger scale and different types of environments. Moreover, we want to extract more salient features that will serve as landmarks. Last but not least, our work is part of a larger project where the navigation strategies are adapted according to the information available. This piece of work is trying to tackle the localisation and map building problem with minimal sensing.

# Appendix B

# Depth Estimation for Autonomous Robot Navigation: A Comparative Approach

Sotirios Ch. Diamantas, Anastasios Oikonomidis, and Richard M. Crowder. IEEE IST 2010, Thessaloniki, Greece, 2010.

**Abstract**

*Depth estimation has long been a fundamental problem both in robotics science and in computer vision. Various methods have been developed and implemented in a large number of applications. Despite the rapid progress in the field the last few years, computation remains a significant issue of the methods employed. In this work, we have implemented two different strategies for inferring depth, both of which are computationally efficient. The first one is inspired by biology, that is optical flow, while the second one is based on a least squares method. In the first strategy, we observe the length variation of the optic flow vectors of a landmark at varying distances and velocities. In the second strategy, we take snapshots of a landmark from different positions and use a least squares approach to estimate the distance between the robot and a landmark. An evaluation of the two different strategies for various depth estimations has been deployed and the results are presented in this paper.*

## B.1 Introduction

Depth estimation is an important task for a large number of robot navigation techniques. In this work, we estimate the distance between a robot and a landmark with the view to support the localisation and mapping problem, a problem which is at the heart of today's mobile robotic systems. This type of work has been developed in order to be integrated into a self-localisation system as is in Diamantas and Crowder (2009) or similar type of systems as described in Amigoni et al. (2004); Einsele (1997). These type of systems make no use of dead reckoning or odometry information and the position of the robot is estimated by means of a laser range finder. Thus, for a robot to be able to estimate its position in an environment, corners and other related features are extracted from the laser range scans. There are, however, cases where a robot navigates in an environment that does not contain these type of features. Such type of environments are the corridors or environments without geometrical objects within the range of the laser scanner. In this paper, the distance of the robot to a landmark is used as a means to calculate the translational distance the robot has covered in that environment while there is no presence of corner features. We, therefore, have adopted two different strategies for inferring the distance between the robot and a landmark and compare the results from both approaches. The first one, that is optical flow, has been inspired by biology and the way insects, in particular honeybees, navigate. The second one is a least squares method that calculates the position of a landmark based on different snapshots taken during the navigation process of the robot.

Optical flow, that is the rate of change of image motion in the retina or a visual sensor, is extracted from the motion of an agent. Optical flow has extensively been used thus far for obstacle avoidance and centring a robot in corridor-like environments Camus et al. (1996); Warren and Fajen (2004); Merrell et al. (2004); Hrabar et al. (2005). Optical flow for depth estimation has been used in Slesareva et al. (2005); Satoru et al. (1999) and a least squares approach for inferring depth appears in Malik and Choi (2009); Boley et al. (1995). Furthermore, a large amount of effort has been focussed on using total least squares Bab-Hadiashar and Suter (1997); jen Tsai et al. (1998), least squares Maybank (1986), or constrained total least squares methods jen Tsai et al. (1999) for calculating the optical flow field.

This paper is comprised of five sections. The following section, Section II, presents the background work while Section III presents the methodology of the two different strategies for depth estimation. Section IV presents and discusses the results from the two strategies. Finally, Section V epitomises the paper with a discussion on the conclusions drawn from this work.

## B.2 Background Work

This section describes the mathematics that underlie the Lucas and Kanade (1981) (LK) algorithm which has been employed in this research work. In order for the optical flow algorithms to perform well, some suitable images need to be chosen. This suitability refers to images that have high texture and contain a multitude of corners. Such images have strong derivatives and, when two orthogonal derivatives are observed then this feature may be unique, and thus, good for tracking. Tracking a feature refers to the ability of finding a feature of interest from one frame to a subsequent one. Tracking the motion of an object can give the flow of the motion of the objects among different frames. In Lucas-Kanade algorithm corners are more suitable than edges for tracking as they contain more information. For the implementation of the LK algorithm the OpenCV (2008) library has been used.

The optical flow algorithm of Lucas-Kanade presupposes three main criteria to produce satisfactorily results. These are:

I. Brightness constancy. The brightness of a pixel does not change from frame to frame, that is $I(x, y, t) = I(x + u, y + v, t + 1)$.

II. Temporal persistence or small movements. The motion of the object that is tracked moves smoothly from frame to frame, that is $I_x u + I_y v + I_t = 0$, where $v, u$ are the $x, y$ components of the velocity $\vec{u}$.

III. Spatial coherence. Neighbouring points of a pixel that belong to the same surface have typically similar motion, and project to nearby points on the image plane.

The equation in the second criterion is an under constrained equation since it involves two unknowns for any given pixel and cannot be used to solve the motion of a pixel in the two dimensions. For this reason the third criterion is used as an assumption to solve the full motion of a pixel in the two dimensions. The third criterion assumes that the neighbouring pixels of any given pixel move coherently as they belong to the same object and project to the same image plane as the given pixel projects. Thus, for tackling the problem in case, the brightness values of the neighbouring pixels are considered and solve a system of linear equations Bradski and Kaehler (2008). Hence, if we take a window of $5 \times 5$ pixels a system of 25 linear equations needs to be solved. However, if the window is too small the *aperture problem* may be encountered where only one dimension of the motion of a pixel can be detected and not the two-dimensional. On the other hand, if the window is too large then the spatial coherence criterion may not be met. Nevertheless, the system that needs to be solved following a window of $5 \times 5$ pixels is expressed by (B.1)

$$\underbrace{\begin{bmatrix} I_x(p1) & I_y(p1) \\ I_x(p2) & I_y(p2) \\ \vdots & \\ I_x(p25) & I_y(p25) \end{bmatrix}}_{A = 25 \times 2} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u} = 2 \times 1} = - \underbrace{\begin{bmatrix} I_t(p1) \\ I_t(p2) \\ \vdots \\ I_t(p25) \end{bmatrix}}_{b = 25 \times 1} \tag{B.1}$$

The goal on the above system of linear equations is to minimise $||A\vec{u}-b||^2$ where $A\vec{u}=b$ is solved by employing least-squares minimisation as in (B.2),

$$(A^T A)\vec{u} = A^T b \tag{B.2}$$

where $A^T A, \vec{u}$, and $A^T b$ are equal to (B.3),

$$\underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}}_{A^T A} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u}} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b} \tag{B.3}$$

and the solution to the equation is given by (B.4)

$$\vec{u} = \begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b. \tag{B.4}$$

If $A^T A$ is invertible, i.e., no zero eigenvalues, it means it has full rank 2 and two large eigenvectors. This occurs in images where there is high texture in at least two directions. If the area that is tracked is an edge, then $A^T A$ becomes singular, that is (B.5),

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} -I_y \\ I_x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{B.5}$$

where $-I_y, I_x$ is an eigenvector with eigenvalue 0. If the area of interest is homogeneous then $A^T A \approx 0$ implying 0 eigenvalues. The pyramidal approach of the LK algorithm overcomes the local information problem at the top layer by tracking over large spatial scales and then as it proceeds downwards to the lower layers the velocity criteria are refined until it arrives at the raw image pixels.

## B.3 Methodology

This section describes the methodology followed for estimating depth using optical flow and least squares. For the optical flow strategy we used the 3D simulation environment in Klein (2002), while for the least squares strategy we employed the Gerkey et al. (2003) 2D simulator. The programming environment of MATLAB (2005) was used for the analysis of the data. The purpose of both methods is to estimate the translational distance a robot has covered in a corridor-like environment due to the absence of ‘distinct’ features in laser scans. The translational distance a robot has travelled can be calculated using the trigonometric functions once the position of a landmark has been estimated.

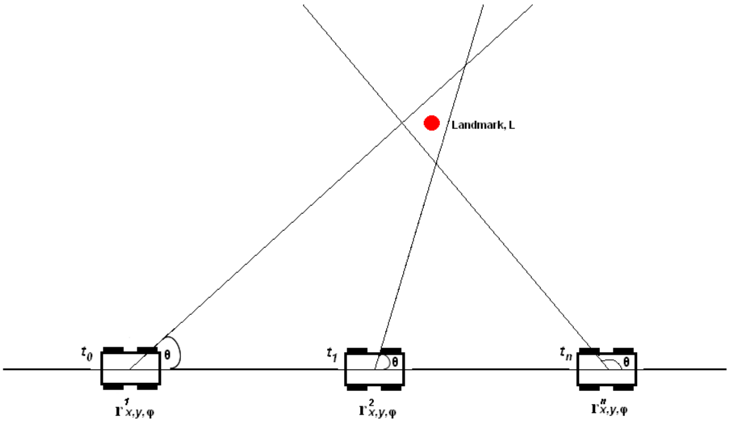


FIGURE B.1: Robot snapshots of the landmark $L$ taken at different time steps $t$ and at different robot positions $r$.

### B.3.1 Least Squares

For the estimation of depth using the least squares approach we track a visual landmark in the environment and infer the position of the landmark based on the orientation and position of the robot. A similar approach has been used by Boley et al. (1995) to infer the starting position of the robot.

In this simulation experiment, we have taken various snapshots of a landmark at equally spaced time steps and measure the angle $\theta$ between the robot position and the visual landmark. Every snapshot represents a linear equation and the solution of the linear system gives us the position of the landmark $L$. Thus, we form a set $\Omega_i$ for every snapshot taken. Equations (B.6) and (B.7) show an example of two snapshots,

$$L \in \Omega_1 = \{h \in \Re^2 | \underbrace{(v_1 - r_1)^T}_{\alpha_1} h = \underbrace{v_1^T \cdot r_1 - ||r_1||^2\}}_{\beta_1} \tag{B.6}$$

$$L \in \Omega_2 = \{h \in \Re^2 | \underbrace{(v_2 - r_2)^T}_{\alpha_2} h = \underbrace{v_2^T \cdot r_2 - ||r_2||^2\}}_{\beta_2} \tag{B.7}$$

where $r_1$ and $r_2$ are the positions of the robot, and $v_1$ and $v_2$ are any two points on a line which is perpendicular to the line formed between the robot and the landmark, that is angle $\theta$. In essence, what we are trying to find is the $\Omega_1 \cap \Omega_2 = \emptyset$. Figure B.1 shows a robot at three different positions, $r^1$, $r^2$, and $r^n$ at corresponding time intervals $t_0$, $t_1$, and $t_n$, that also represent three different snapshots.

The following equations, (B.8) and (B.9), show the process for a $n$ number of snapshots. Noise in the system is represented by variable $\epsilon_i$.

$$
\begin{aligned}
h\alpha_1 + \epsilon_1 &= \beta_1 \\
h\alpha_2 + \epsilon_2 &= \beta_2 \\
&\vdots \\
h\alpha_n + \epsilon_n &= \beta_n
\end{aligned}
\tag{B.8}
$$

$$
h \in argmin \sum_{i=1}^{n} (h\alpha_i - \beta_i + \epsilon_i)^2 \tag{B.9}
$$

$$
\underbrace{\left( \sum_{i=1}^{n} \alpha_i \alpha_i^T + \epsilon_i \right)}_{C} h = \underbrace{\left( \sum_{i=1}^{n} \alpha_i \beta_i \right)}_{\gamma} \tag{B.10}
$$

$$
h = C^{-1}\gamma \tag{B.11}
$$

In (B.10) $C$ is a $2 \times 2$ matrix and $\gamma \in \Re^2$. The position of the landmark is, thus, given by $h$ where we are interested in its y-axis element, that is depth. In this experiment we have used $n = 1000$ observations (also explained in the next subsection) with varying distances between the landmark and the robot. We have performed 10 trials for each observation in order to estimate the mean position of the landmark. In addition, the error, $\epsilon_i$, in $\theta$ is uniformly distributed and it has been tested with three different deviations $\pm 1^\circ$, $\pm 3^\circ$, and $\pm 5^\circ$.

### B.3.2 Optical Flow

In order for the robot to localise in an environment using optical flow vectors, a training data set of $n = 1000$ observations has been implemented where a vector is observed at varying distances between the robot and the landmark, and at varying velocities. Moreover, the distances and velocities chosen to create the training set approximate the real distributions of distance and velocity when a robot navigates in an environment. Thus, a joint probability distribution has been created by two continuous and independent variables, that is distance, $D$, and velocity, $E$, and is expressed by (B.12),

$$
f_{D,E}(d, e) = f_D(d) \cdot f_E(e) \qquad \forall d, e. \tag{B.12}
$$

The velocity (in $km/h$) and the distance (in $metres$) variables have been drawn from two Gaussian distributions with $\mu = 4$, $\sigma = 1$ and $\mu = 11$, $\sigma = 3$, respectively. The $n$ observations model the position of the optic flow vector in the plane under $n$ varying distances and velocities. Figure B.2 provides a pictorial representation of Gaussian distributions for velocity and distance.

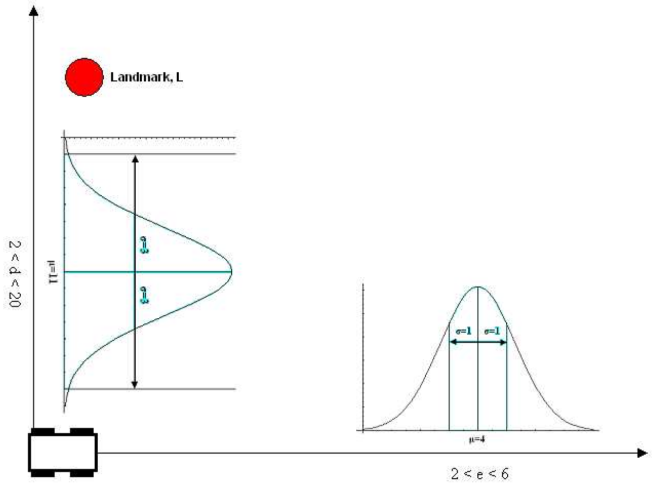


Figure B.2: Gaussian distributions used for modelling distance, $D$, and velocity, $E$, in the optical flow strategy. Their mean and standard deviation are $\mu = 11$, $\sigma = 3$ and $\mu = 4$, $\sigma = 1$, respectively. The range of their values is $2 < d < 20$ $(metres)$ for distance and $2 < e < 6$ $(km/h)$ for velocity.

One assumption that needs to be met in our method is that the majority of the vectors comprising a given landmark should have the same, or almost the same magnitude. In addition, the orientation of the camera on the robotic platform is perpendicular to the direction of motion so as a translational optical flow information is generated. During the testing of the optical flow algorithm the velocity of the robot is not known. The only information employed for inferring depth is the optical flow vectors of the landmark.

Regression analysis has been employed to estimate the distance between a landmark and the robot based only on the observed length of the optical flow vectors. The regression formula that expresses this distance is described by (B.13),

$$\ln D_i = a + b \cdot len_i + \epsilon_i \tag{B.13}$$

where $\ln D_i$ is the expected distance, $a$ is the constant, $b$ is the coefficient, $len_i$ is the length of the optical flow vectors, and $\epsilon_i$ is the error. The following section presents the results from the two strategies.

## B.4 Results

In the least squares strategy, as it would be expected, the fit of the estimated distance to the actual distance increases as the number of snapshots increase and as the mean simulation error decreases. However, the larger the number of snapshots the greater the computational cost. Table B.1 presents the $R^2$ statistic for various combinations of number of snapshots and mean simulation error. This $R^2$ statistic for $n$ snapshots and different error corresponds to the fit of the model $D = 0 + 1 \cdot E(D)$ where $D$ is the actual distance and $E(D)$ is the expected distance estimated through the least squares method. The minimum number of snapshots is 4 as this is the number required for the system to become overdetermined. Assuming that estimated distance is an unbiased estimate of the actual distance, if linear regression is used to estimate (for each combination of number of snapshots and mean simulation error) $a$ and $b$ in the function $D = a + b \cdot E(D)$, using the experiment's data, it should turn out that the best estimates are (approximately) $a = 0$ and $b = 1$. However, this is not the case as there seems to be a tendency for this method to systematically underestimate distance, especially when the mean simulation error is high and the number of snapshots low. The level of underestimation is more obvious if in the previous function the constant is constrained to be 0. Interestingly, even though the bias decreases as the number of snapshots increases or the mean simulation error decreases, it is always significant as the standard error of the coefficient decreases too. In the following Table B.2 the estimated coefficient of expected distance is presented for each combination of number of snapshots and mean simulation error. The confidence interval is 95%.

Table B.1: Performance of Least Squares method - $R^2$

| Snapshots $n$ | $R^2$ ($\epsilon_i \pm 1°$) | $R^2$ ($\epsilon_i \pm 3°$) | $R^2$ ($\epsilon_i \pm 5°$) |
|---|---|---|---|
| 4 | 0.971 | 0.137 | 0.000 |
| 6 | 0.991 | 0.722 | 0.000 |
| 8 | 0.996 | 0.883 | 0.336 |
| 10 | 0.998 | 0.940 | 0.649 |
| 12 | 0.999 | 0.967 | 0.801 |
| 14 | 0.999 | 0.979 | 0.881 |
| 16 | 1.000 | 0.987 | 0.915 |
| 18 | 1.000 | 0.991 | 0.944 |
| 20 | 1.000 | 0.993 | 0.959 |

Table B.2: Performance of Least Squares method - Coefficient

| Snapshots $n$ | Coeff. ($\epsilon_i \pm 1°$) | Coeff. ($\epsilon_i \pm 3°$) | Coeff. ($\epsilon_i \pm 5°$) |
|---|---|---|---|
| 4 | 1.017 | 1.166 | 1.726 |
| 6 | 1.013 | 1.130 | 1.386 |
| 8 | 1.009 | 1.084 | 1.239 |
| 10 | 1.006 | 1.057 | 1.162 |
| 12 | 1.004 | 1.042 | 1.118 |
| 14 | 1.004 | 1.033 | 1.089 |
| 16 | 1.003 | 1.026 | 1.073 |
| 18 | 1.002 | 1.021 | 1.059 |
| 20 | 1.002 | 1.018 | 1.049 |

In the optical flow strategy, after experimenting with various functional forms, we concluded that modelling the natural logarithm of distance as a linear function of the observed vectors' length is the optimal method to estimate distance. Thus, linear regression is utilised to estimate the combination of $a$ and $b$ that minimises the squared error in (B.13). Equation (B.13) has also been employed to compute $R^2$ at different velocity intervals in order to adapt to realistic implementations. The outputs of these regression models are presented in Table B.3.

TABLE B.3: Performance of Optical Flow method

| Velocity $e$ ($km/h$) | Constant $a$ | Coefficient $b$ | $R^2$ |
|---|---|---|---|
| $2 < e < 6$ | 2.96 | -0.075 | 0.6029 |
| $2 < e < 5$ | 3.03 | -0.09 | 0.6483 |
| $2 < e < 4$ | 3.08 | -0.11 | 0.7569 |
| $2 < e < 3$ | 3.1 | -0.143 | 0.8052 |

Obviously, velocity influences the length of the optic flow vectors and the smaller the range used the better the estimation of the expected distance. However, we avoided using velocity as a variable in the tests as this work is intended to be used in self-localisation systems where the position of the robot is provided only by means of a laser range finder. Hence, velocity cannot be estimated with a laser finder.

The results show that optical flow can perform better in cases where the mean error is high and a small number of snapshots have been taken in the least squares strategy. Nevertheless, when velocity range is large enough, optical flow does not perform satisfactorily enough when compared to least squares.

## B.5 Conclusion

In this work we have addressed the problem of depth estimation using two different strategies, namely optical flow and least squares. The idea is to use a depth estimation method in a self-localisation system where the only sensor used for providing odometry data is a laser range finder. Such type of systems cannot operate in environments where there are no 'distinct' features like corners. Thus, the addition of a visual sensor can support the localisation process of the robot.

The results in this paper reveal that least squares is conditional on the number of snapshots and the mean simulation error. Optical flow, on the other hand, is conditional on the velocity of the robot. The results presented in this paper show that there is a trade-off between accuracy and computational cost.

## Acknowledgment

The authors would like to thank Dr Renato Cavalcante for his helpful comments on this work.

# Appendix C

# Towards Optical Flow-based Robotic Homing

Sotirios Ch. Diamantas, Anastasios Oikonomidis, and Richard M. Crowder. IJCNN 2010 (IEEE WCCI 2010), Barcelona, Spain, 2010.

**Abstract**

*This paper presents a novel biologically-inspired approach for tackling the problem of robot homing. In our method the only information employed is optical flow. Optical flow, which is not a property of landmarks like colour, shape, and size but a property of the camera motion, is used for localising an autonomous robot in* a priori *unknown environment. Our method exploits the optical flow 'fingerprint' of landmarks caused by the motion of the robot in the environment. For this purpose, we have developed a training algorithm that estimates the probability of observing the same landmark from varying distances and velocities. Our method promises to be computationally efficient and inexpensive. The simulation results we present show the validity of our methods.*

## C.1 Introduction

Visual navigation lies at the heart of mobile robotics. Homing (or inbound journey) refers to the navigation process where an autonomous agent performs a return to its home position after having completed foraging (or outbound journey; foraging is mainly attributed to a biological agent). A robot may have to return to its base for a number of reasons like recharging batteries, failure of a subsystem, or completion of a task. The application areas of robots capable of performing homing are plenty and vary. Search and rescue robots are in need in areas that have been hit by earthquakes or in environments that are hazardous for humans Matsuno and Tadokoro (2004). Planetary missions to other regions constitute another application area of robots whose navigation process involves returning back to their base. In this paper we have developed a novel approach to tackle the problem of robot homing using visual modality as the only source of information. No other sensor is provided to the robotic agent apart from two side-ways cameras mounted on a simulated mobile platform.

Optical flow, that is the rate of change of image motion in the retina or a visual sensor, is extracted from the motion of the autonomous agent. The orientation of the cameras on the robotic platform are perpendicular to the direction of motion so as a translational optic flow information is generated. Optic flow, which is not a property of the landmarks, like colour, shape, and size, but a property of the camera motion has been used for building topological maps in *a priori* unknown environment based on the optical flow patterns of the landmarks. The novelty of our method lies in the fact that no information is given such as the position or the velocity of the robot but only the optical flow 'fingerprint' of the landmarks caused by the motion of the robot. For this purpose, a training algorithm has been deployed and a probability is inferred that is computed from the similarity of the optical flow patterns between the outbound and inbound journeys.

Biology is seen as an alternative solution to the problems robots encounter which includes algorithmic complexity, performance, and power consumption among others. Biological inspiration provides simple, yet effective methods for the solutions of such problems. The careful examination of those methods has twofold gain. The study of the principles of biological organisms entails making better autonomous systems that will, in turn, help us perceive and understand better the underlying mechanisms that underpin the biological organisms.

This paper comprises five sections. Following is Section II where related as well as background work is presented. In Section III the methodology of the homing model is described. Section IV presents the results of the statistical model on the homing process of navigation. Finally, Section V epitomizes the conclusions drawn from this work and indicates a number of areas that further research is attainable.

## C.2 Related Work

A large number of insects use optic flow for navigation. Insects like *Drosophila* use the apparent visual motion of objects to supply information about the three-dimensional structure of the environment. The fly *Drosophila* uses optic flow to pick near targets. Collett in Collett (2002) shows that in insects the task of evaluating distances between objects is made easier by making side-to-side movements of the head strictly translational and disregarding any rotational components that can influence the distance to the objects. Looming, i.e., image expansion, can also distort the actual distance to the object as the apparent size compared to the physical size of the object differs. Collett (2002) in his experiments ascertains that *Drosophila* like many insects limit rotational flow during exploratory locomotion. In fact, *Drosophila* move in straight-line segments and restrict any rotation to saccades at the end of each segment. Schuster et al. Schuster et al. (2002) have used virtual reality techniques to show that fruit flies use translational motion for picking up the nearest object while disregarding looming.

Ladybirds also move in straight-line segments and rely on translational optic flow rather that looming cues. Other animals like locusts and mantids turn their head from one side to the other just before jumping. Kral and Poteser Kral and Poteser (1997) suggest that locusts and mantids use translational motion to infer the three-dimensional structure of the environment and in particular the distance to the object they wish to approach. In some other experiments performed by Tautz et al. Tautz et al. (2004) trained bees had to travel large distances across various scenes that included both land and water. The results showed that the flights over water had a significantly flatter slope than the ones above land. This suggests that the perception of distance covered by bees is not absolute but scene-dependent where the optic flow perceived is evidently larger. This may also suggest why some bees are drowning by 'diving' into lakes or the sea while flying above water. The distance and direction to a food source is communicated in the bees by means of waggle dances that integrate retinal image flow along the flight path Esch et al. (2001), von Frisch (1993).

Two well-known homing models are the *snapshot* and the *Average Landmark Vector* (ALV) model. The *snapshot* model is an implementation of the template hypothesis Cartwright and Collett (1983), Cartwright and Collett (1987a). It requires a panoramic snapshot of the goal position, be it a hive, nest, or a food source. Along with the snapshot the compass direction is stored. The *snapshot* model is an image matching process between a snapshot taken at a goal position and a snapshot containing the current view. The image obtained from the omnidirectional camera is unwrapped and a threshold operation is performed to yield a one-dimensional black and white image. The landmarks are denoted as black marks on the image. Then, this is compared with the snapshot of the current view to produce the homing vector. The *homing vector* is a two-dimensional vector pointing towards the home position and is obtained by summing up all radial and tangential vector components. The ALV model Lambrinos et al. (2000) uses, too, a processed panoramic image but, in contrast to the *snapshot* model, it need not be stored. Only a two-dimensional vector for each landmark needs to be stored that points to the direction of the landmark. Matching and unwrapping of the image are not required since the calculations are performed on the basis of vector components. Thus, ALV is more parsimonious than the *snapshot* model. Nevertheless, snapshots in the ALV model have to be captured and processed to produce a one-dimensional picture, as is in the snapshot model. A compass information is required for the ALV model as well.

The *snapshot* and the *Average Landmark Vector* are two models that have been inspired by the way insects perform homing. On one hand, their main advantage is the simplicity of the method that entails a low computational complexity. On the other hand, their disadvantages are that both methods are applied at the end of the homing process, that is, when an agent is close to its home position. Moreover, all landmarks need to be visible both in the current snapshot as well as in the stored snapshot, that is, they must be the same landmarks in both snapshots. In addition to the biologically inspired methods, other methods of homing have been developed that make use of the Extended Kalman Filter (EKF) or homing methods that are based on panoramic vision Argyros et al. (2005). The former are trying to tackle the Simultaneous Localisation and Mapping (SLAM) problem and make use of probability to build accurate maps that are based on vehicle position estimates. In turn, these maps provide a more accurate estimate of the vehicle's position.

A work by Newman et al. (2002) addresses the problem of homing through the localisation and mapping method using a laser scanner and wheel encoders. In their method they extract line segments from a laser scanner that act as features while the position of the robot is calculated from wheel encoder readings. A major issue of the SLAM methodology lies in the accumulation of errors as the robot navigates. The slippage of the terrain causes erroneous sensor readings and, the harsher a terrain is, the larger the accumulation of errors that occur. The main disadvantage of SLAM, however, is the computational cost that increases quadratically as new observations are made by the sensor be it a laser scanner or a camera Smith et al. (1988); Davison (2003). Updating all features and the joint covariance matrix each time a new observation is made adds significantly to the computational cost of the method. Another issue in the SLAM methodology is the *data association* problem where features or landmarks look alike. FastSLAM uses Rao-Blackwellised particle filters and is a method that tries to alleviate the *data association* problem Hahnel et al. (2003); Montemerlo et al. (2002). SLAM methods can be applied to a wide range of environments, such as indoors, outdoors, dynamic, or large-scale environments. It can, thus, be understood why SLAM has risen to one of the most research-intensive problems in the robotics field.

### C.2.1 Applications of Optical Flow

Lately a growing number of autonomous vehicles have been built using techniques inspired by insects and, in particular, optical flow. One of the first works that studied the relation of scene geometry and the motion of the observer was by Gibson (1974). A large amount of work, however, has been focussed on obstacle avoidance using optical flow Camus et al. (1996); Warren and Fajen (2004); Merrell et al. (2004). The technique, generally, works by splitting the image (for single camera systems) into left- and right-hand side. If the summation of vectors of either side exceeds a given threshold then the vehicle is about to collide with an object. Similarly, this method has been used for centring autonomous robots in corridors or even a canyon Hrabar et al. (2005) with the difference that the summation of vectors this time must be equal in both the left-hand side and the right-hand side of the image. Ohnishi and Imiya (2007) utilise optical flow for both obstacle avoidance and corridor navigation. The performance of optical flow has also been tested in underwater colour images by Madjidi and Negahdaripour (2006). Vardy (2005) deploys various optical flow techniques which are compared using block matching and differential methods to tackle homing.

In a recent work implemented by Kendoul et al. (2009) optic flow is used for a fully autonomous flight control of an Unmanned Aerial Vehicle (UAV). The distance travelled in this UAV is calculated by integrating the optical flow over time. A similar work for controlling a small UAV in confined and cluttered environments has also been implemented by Zufferey et al. (2008). Barron et al. (1994) discuss the performance of optical flow techniques. Their comparison is focussed on accuracy, reliability and density of the velocity measurements. Other works employ optic flow methods for depth perception Simpson (1993), motion segmentation Blackburn and Nguyen (1995), or estimation of ego-motion Frenz and Lappe (2005).

A similar technique to optical flow developed by Langer and Mann (2003) called *optical snow* arises in situations where camera motion occurs in highly cluttered 3D environments. Such cases involve a passive observer watching the fall of the snow, hence, the name of the method. *Optical snow* has been inspired by research in animals that inhabit in highly dense and cluttered environments; such animals include the rabbit, the cat, and the bird. The properties of the *optical snow* are that yields dense motion parallax with many depth discontinuities occurring in almost all image points. This comes in contrast to the classical methods that compute optical flow and presuppose temporal persistence and spatial coherence. In the work of Langer and Mann (2003) the properties of *optical snow* in the Fourier domain are presented and investigate its computational problems on motion processing.

### C.2.2 Mathematical Foundations of Optical Flow

This section describes the mathematics that underlie the optic flow algorithms, and in particular, the Lucas and Kanade (1981) (LK) algorithm that has been employed in this research work. In order for the optic flow algorithms to perform well, some suitable images need to be chosen. This suitability refers to images that have high texture and contain a multitude of corners. Such images have strong derivatives and, when two orthogonal derivatives are observed then this feature may be unique, and thus, good for tracking. Tracking a feature refers to the ability of finding a feature of interest from one frame to a subsequent one. Tracking the motion of an object can give the flow of the motion of the objects among different frames. In Lucas-Kanade, algorithm corners are more suitable than edges for tracking as they contain more information. For the implementation of the LK algorithm the OpenCV (2008) library has been used.

The optic flow algorithm of Lucas-Kanade presupposes three main criteria to produce satisfactorily results. These are:

I. Brightness constancy. The brightness of a pixel does not change from frame to frame, that is $I(x, y, t) = I(x + u, y + v, t + 1)$.

II. Temporal persistence or small movements. The motion of the object that is tracked moves smoothly from frame to frame, that is $I_x u + I_y v + I_t = 0$, where $v, u$ are the $x, y$ components of the velocity $\vec{u}$.

III. Spatial coherence. Neighbouring points of a pixel that belong to the same surface have typically similar motion, and project to nearby points on the image plane Bradski and Kaehler (2008).

The equation in the second criterion is an under constrained equation since it involves two unknowns for any given pixel and cannot be used to solve the motion of a pixel in the two dimensions. For this reason the third criterion is used as an assumption to solve the full motion of a pixel in the two dimensions. The third criterion assumes that the neighbouring pixels of any given pixel move coherently as they belong to the same object and project to the same image plane as the given pixel projects. Thus, for solving the problem in case, the brightness values of the neighbouring pixels are taken and solve a system of linear equations Bradski and Kaehler (2008). Hence, if we take a window of 5 × 5 pixels a system of 25 linear equations needs to be solved. However, if a window is too small the *aperture problem* may be encountered where only one dimension of the motion of a pixel can be detected and not the two-dimensional. On the other hand, if a window is too large then the spatial coherence criterion may not be met. Nevertheless, the system that needs to be solved following a window of 5 × 5 pixels is expressed by (C.1)

$$\underbrace{\begin{bmatrix} I_x(p1) & I_y(p1) \\ I_x(p2) & I_y(p2) \\ \vdots & \\ I_x(p25) & I_y(p25) \end{bmatrix}}_{A = 25 \times 2} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u} = 2 \times 1} = - \underbrace{\begin{bmatrix} I_t(p1) \\ I_t(p2) \\ \vdots \\ I_t(p25) \end{bmatrix}}_{b = 25 \times 1} . \tag{C.1}$$

The goal on the above system of linear equations is to minimise $||A\vec{u} - b||^2$ where $A\vec{u} = b$ is solved by employing least-squares minimisation as in (C.2),

$$(A^T A)\vec{u} = A^T b \tag{C.2}$$

where $A^T A, \vec{u}$, and $A^T b$ are equal to (C.3),

$$\underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}}_{A^T A} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{u}} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b} \tag{C.3}$$

and the solution to the equation is given by (C.4)

$$\vec{u} = \begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b. \tag{C.4}$$

If $A^T A$ is invertible, i.e., no zero eigenvalues, it means it has full rank 2 and two large eigenvectors. This occurs in images where there is high texture in at least two directions. If the area that is tracked is an edge, then $A^T A$ becomes singular, that is (C.5),

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} -I_y \\ I_x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{C.5}$$

where $-I_y, I_x$ is an eigenvector with eigenvalue 0. If the area of interest is homogeneous then $A^T A \approx 0$ implying 0 eigenvalues. The reason that the LK algorithm was chosen is that it is a fast and accurate optical flow algorithm. It relies on local information rather than global as is the nature of Horn and Schunck (1981) algorithm. This has the advantage of performing fast optic flow operations. Nevertheless, the disadvantage of local information, that is derived from a small window, is that large motions can move out of the local window. The pyramidal approach of the LK algorithm uses a coarse-to-fine iterative method, that is, various layers in scale-space to overcome the local information problem. Thus, the optical flow problem is first solved at the top layer by tracking over large spatial scales and then as it proceeds downwards to the lower layers the initial velocity criteria are refined until it arrives at the raw image pixels.

## C.3 Methodology

In this section we describe how the optical flow ‘signature’ of the landmarks, that is caused by the perceived motion of the robot in the environment, can be used to localise the robot during the homing process. Various landmarks have been modelled and simulated from which the robot passes through. The simulated landmarks have geometrical shapes like a sphere or a rectangular and they are textured in order to produce large amounts of optic flow (as is in real environments). As mentioned in Section I the simulated robot consists of two side-ways cameras which are perpendicular to the direction of motion. This creates a translational optic flow as the robot navigates through the environment. Every landmark in the environment ‘emits’ a number of optic flow vectors that are dependent on the distance between the robot and the landmark, and the velocity of the robot. One of the advantages of our method is that images are only captured and are not used for storage or comparison. Storing and comparing only the properties of vectors between different frames, that is, the mean position of all the vectors and the number of vectors, reduces the computational complexity and the cost of the homing process.

During the outbound trip of the robot the camera calculates and stores the optic flow vectors that are generated by the motion of the vehicle. During this phase the robot builds a topological map from the optical flow ‘fingerprint’ of the landmarks. After the foraging trip has completed the homing trip is initiated. In the homing phase, the robot compares the optical flow patterns it currently perceives with the ones occurred during the foraging journey. If the similarity score (i.e., probability) between the two patterns is above a given threshold, then the robot assumes the current landmark observed is the same with the landmark observed during the outbound trip. This information is then used to localise the robot within the topological map. The similarity score of the vectors is a probabilistic result of the Euclidean distance of the vectors between the current image and the image taken during the outbound journey.

In order for the robot to localise in an environment using optic flow vectors, a training data set of $n = 1000$ observations has been implemented where a vector is observed at varying distances between the robot and the landmark, and at varying velocities. The distances and velocities chosen to create the training set approximate the real distributions of velocity and distance when a robot navigates in an environment. Thus, a joint probability distribution has been created by two continuous and independent variables, that is velocity, $C$, and distance, $D$, and is expressed by (C.6)

$$f_{C,D}(c,d) = f_C(c) \cdot f_D(d) \qquad \forall c, d. \tag{C.6}$$

The velocity and the distance variables have been drawn from two Gaussian distributions with $\mu = 4, \sigma = 1$ and $\mu = 11, \sigma = 3$, respectively. The $n$ observations model the position of the vector in the plane in a varying combination of distances and velocities. One assumption that needs to be met in our method is that the majority of the vectors comprising a given landmark should have the same, or almost the same magnitude. In order to solve the similarity problem between vectors, the mean, $x_k, y_k$, or centre point of every vector is taken. Thus, summing up all the mean points of the training set and dividing by the number of observations we end up having the mean of the means, $\bar{x}, \bar{y}$, as shown in (C.7)

$$\bar{x}, \bar{y} = \frac{1}{n} \cdot \sum_{k=1}^{n} x_k, y_k \qquad n = 1000. \tag{C.7}$$

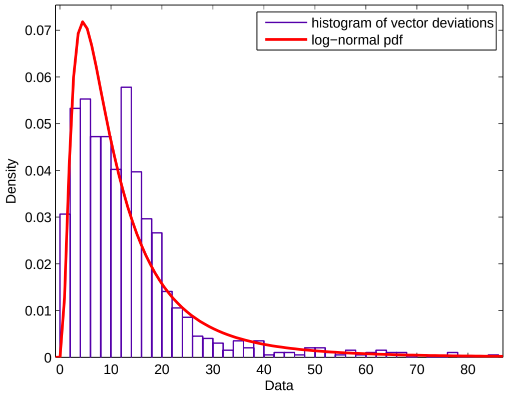


FIGURE C.1: Histogram of vector deviations of the training algorithm and the log-normal probability density function (pdf) fit. Mean and standard deviation are $\mu = 2.24$ and $\sigma = 0.86$, respectively.

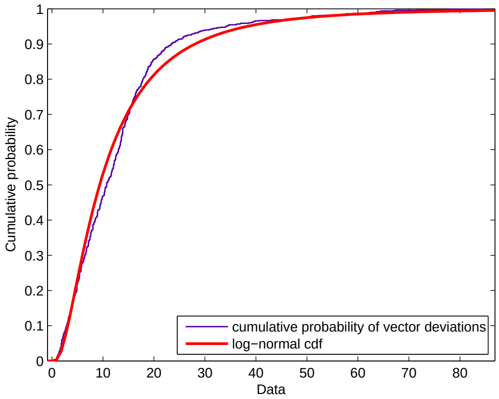


FIGURE C.2: Cumulative density functions (cdf) of vector deviations and the log-normal distribution.

The mean of the means in an optic flow pattern can be visualised as the centre of gravity in a physical system. We then compute the Euclidean distances, $\chi_k$, between the mean of the means and the $n$ observations as expressed by (C.8)

$$\chi_k = \sqrt{(x_k - \bar{x})^2 + (y_k - \bar{y})^2}. \tag{C.8}$$

The histogram produced by the Euclidean distances, $\chi_k$, forms a log-normal probability density function (pdf) with $\mu = 2.24$ in log location and $\sigma = 0.86$ in log scale. Figure C.1 shows the histogram of vector deviations and the probability density function of the log-normal. The log-normal pdf is deployed in order to infer a probability as to how likely it is for the vectors of the current snapshot to have deviated when compared with the vectors of a snapshot stored in memory. Figure C.2 depicts the cumulative distribution function (cdf) of vector deviations and the log-normal.

The cumulative density function of log-normal is expressed by (C.9), where *erfc* is the complementary error function and $\Phi$ is the standard normal cdf. The probability density function of log-normal is given by (C.10)

$$F_X(\delta;\mu,\sigma) = \frac{1}{2} erfc\left[-\frac{\ln\delta-\mu}{\sigma\sqrt{2}}\right] = \Phi\left(\frac{\ln\delta-\mu}{\sigma}\right) \tag{C.9}$$

$$f_X(\delta;\mu,\sigma) = \frac{1}{\delta\sigma\sqrt{2\pi}} e^{-\frac{(\ln\delta-\mu)^2}{2\sigma^2}} \qquad \delta > 0. \tag{C.10}$$

Thus far, we have explained the methodology of the training algorithm. We now move on to the process of calculating a probability for the patterns observed by the robot during the foraging and homing process. This probability will aid the robot localise itself in the environment. During the homing navigation process, the robot calculates the Euclidean distance, $\delta$, between the mean position, $\bar{x}, \bar{y}$, of all the vectors in a given landmark with the mean position of the vectors of the landmarks stored in the database. Equations (C.11), (C.12), and (C.13) describe the process for two distinct landmarks. In (C.11), (C.12), $r$ and $s$ are the number of vectors for two distinct landmarks $i$ and $j$, one of which is observed during the outbound trip while the other one is observed during the inbound trip.

$$\bar{x_i}, \bar{y_i} = \frac{1}{r} \cdot \sum_{a=1}^{r} x_a, y_a \tag{C.11}$$

$$\bar{x_j}, \bar{y_j} = \frac{1}{s} \cdot \sum_{b=1}^{s} x_b, y_b \tag{C.12}$$

$$\delta = \sqrt{(\bar{x_i} - \bar{x_j})^2 + (\bar{y_i} - \bar{y_j})^2} \tag{C.13}$$

$$P = 1 - P_\delta \tag{C.14}$$

The log-normal cdf then gives us the probability $P_\delta$ based on the Euclidean distance $\delta$ between the two sets of vectors. It is then subtracted from 1 to give the probability $P$ as is in (C.14). In addition, the probability $P$ of the log-normal is multiplied by the ratio of the number of the vectors as shown in (C.15) with $min_i$ being the landmark $i$ with the minimum number of vectors and $max_j$ being the landmark $j$ with the maximum number of vectors.

$$P_T = P\left(\frac{min_i}{max_j}\right) \tag{C.15}$$

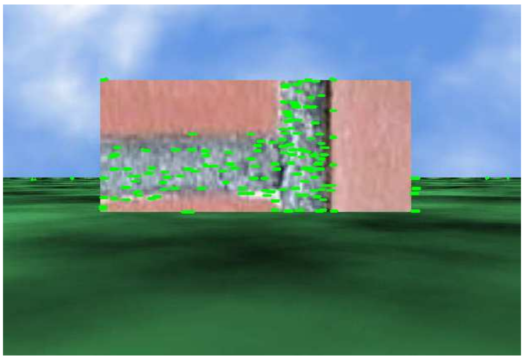


Figure C.3: Snapshot of the reference landmark and its optical flow ‘signature’ taken at a distance of $11m$ and a velocity of $4km/h$.

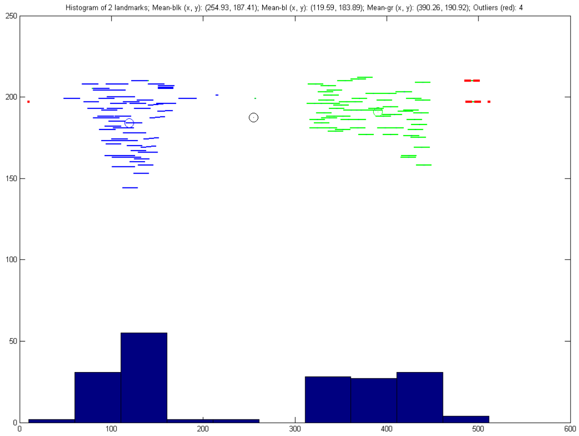


Figure C.4: Clustering of two landmarks by employing the histogram of their vectors.

This results to the total probability (or similarity score), $P_T$. Thus, even if the Euclidean distance, $\delta$, between two sets of vectors is small, the total probability, $P_T$, can be low if the ratio of the vectors is small. Hence, two patterns which are totally different may have a small Euclidean distance that yields a high probability. Multiplying the probability value, $P$, by the ratio of the number of vectors can drop significantly the total probability value, $P_T$, assuming that the number of vectors of the two sets are not of the same multitude. The landmark of Fig. C.3 acts as a reference for the following snapshots in order to demonstrate the similarity score at varying distances and velocities, and between different landmarks. The optic flow images are created by calculating the motion of a landmark between two contiguous frames. It should also be noted that the flow vectors appear upside down since the images are read from top to bottom.

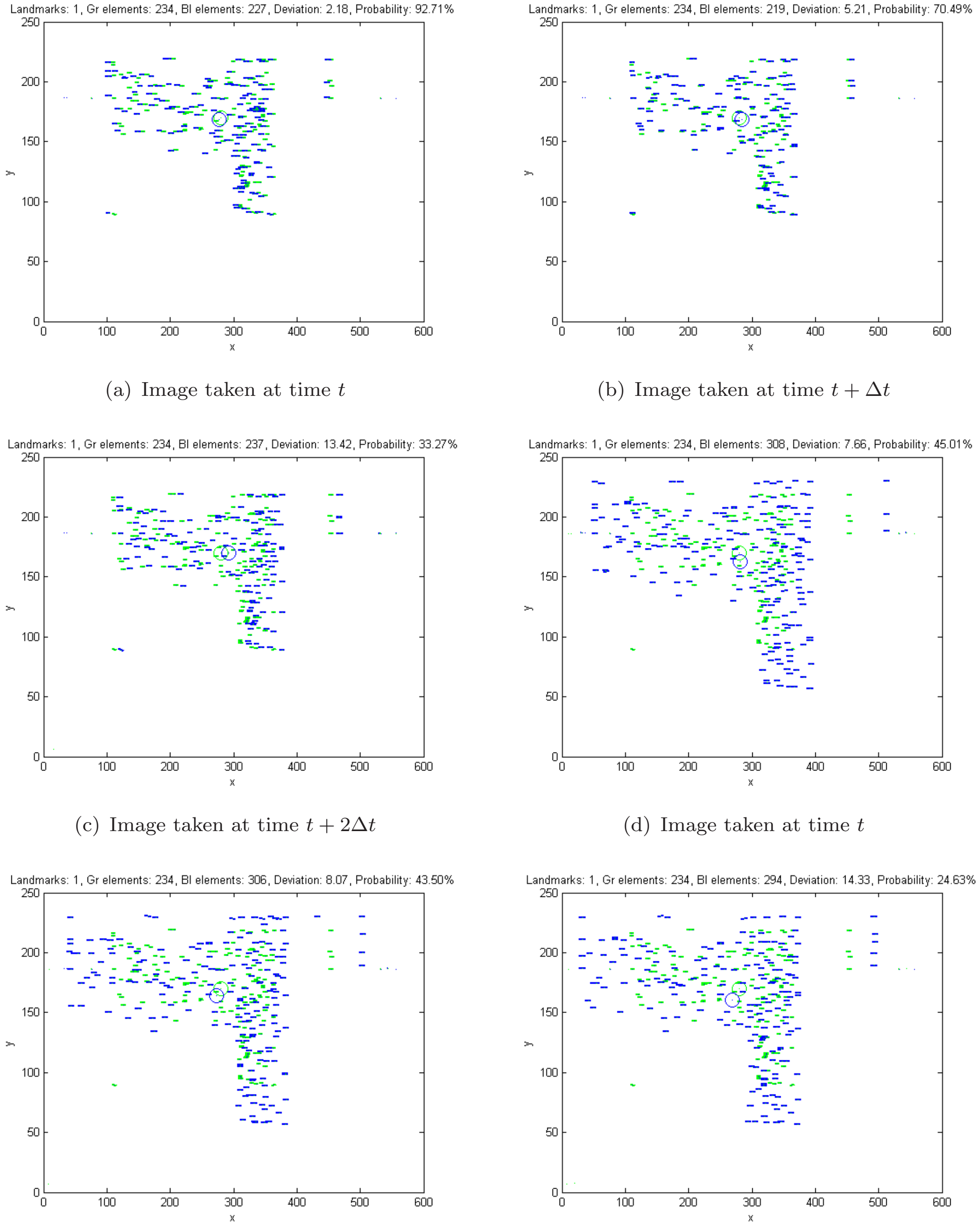
In this work, a clustering algorithm has also been implemented for counting the number of landmarks. The counting of landmarks is a method that is used by insects as revealed in a recent work by Dacke and Srinivasan (2008). Although the clustering algorithm is a rather simple one, distinguishing between different landmarks is an important task. The number of landmarks each one of the two cameras can distinguish is two. Therefore four is the maximum number of landmarks that can be seen at any time. However, in the case where one or more landmarks have gone missing then the robot can still recognise its location from the optical flow ‘fingerprint’ of the remaining ones. The clustering algorithm creates a histogram of the vectors and finds the minimum point between the two peaks of the histogram. Figure C.4 shows an application of the algorithm with two landmarks. In the same figure some outliers have been located, that is, vectors with small length that have been disregarded.

## C.4 Results

The homing model described in this paper has been implemented in C++ programming language and the MATLAB (2005) software has been used for the analysis of the data. The *breve* simulator Klein (2002) was used for the creation of landmarks in 3D. The algorithm was run offline on a Pentium 4 machine at 3.00 GHz with 1.00 GB of RAM. The following graphs, Fig. C.5, demonstrate the effectiveness of our approach by comparing the vectors of the reference image, Fig. C.3, taken at a distance of $11m$ and a velocity of $4km/h$ with the vectors of the same landmark taken at different distances and velocities. Figures C.5(a), C.5(b), and C.5(c) depict the similarity score at a distance of $11m$ and a velocity of $5km/h$ while Figs. C.5(d), C.5(e), and C.5(f) depict the similarity score at a distance of $8m$ and a velocity of $4km/h$.

The circle in the graphs represents the mean position of all the vectors that comprise a landmark. The *green*, (Gr), optic flow vectors refer to the reference image while the *blue*, (Bl), ones refer to the current snapshot. Deviation is the Euclidean distance, $\delta$, between the mean position of the vectors of the current snapshot, $\bar{x}, \bar{y}$, with the mean position of the vectors of the reference image. The number of elements, i.e., vectors, in the current snapshot differs from frame to frame as the angle of perception changes. Time, $t$, denotes the time steps the images were captured. It is clear that the similarity score is quite high in all three images, Figs. C.5(a)-C.5(c). This shows that velocity does not influence significantly the patterns of the images. However, as it is expected, the similarity score drops as time step, $t$, changes, Fig. C.5(b) and Fig. C.5(c). In the remaining graphs, Figs. C.5(d)-C.5(f), of Fig. C.5, the distance at which images were taken is $8m$ while the velocity has been kept the same as is in the reference image, that is, $4km/h$. The similarity score in these figures appears to be lower revealing that distance influences more than velocity the optic flow patterns. Nevertheless, the similarity score can be considered quite satisfactory, in general.

Figure C.6 depicts two different landmarks and the similarity score is inferred against the reference landmark of Fig. C.3. The distance and velocity at which they were captured remains the same as is in the reference image. In the first graph, Fig. C.6(c), the similarity score is quite low, that is 2.04% while in the next graph, Fig. C.6(d), the



(a) Image taken at time $t$

(b) Image taken at time $t + \Delta t$

(c) Image taken at time $t + 2\Delta t$

(d) Image taken at time $t$

(e) Image taken at time $t - \Delta t$

(f) Image taken at time $t - 2\Delta t$

FIGURE C.5: Optical flow vectors of the reference landmark at its initial setting (distance=$11m$, velocity=$4km/h$) against the optical flow vectors of the same landmark taken at a distance of $11m$ and a velocity of $5km/h$, Figs. C.5(a), C.5(b), C.5(c), and at a distance of $8m$ and a velocity of $4km/h$, Figs. C.5(d), C.5(e), C.5(f).

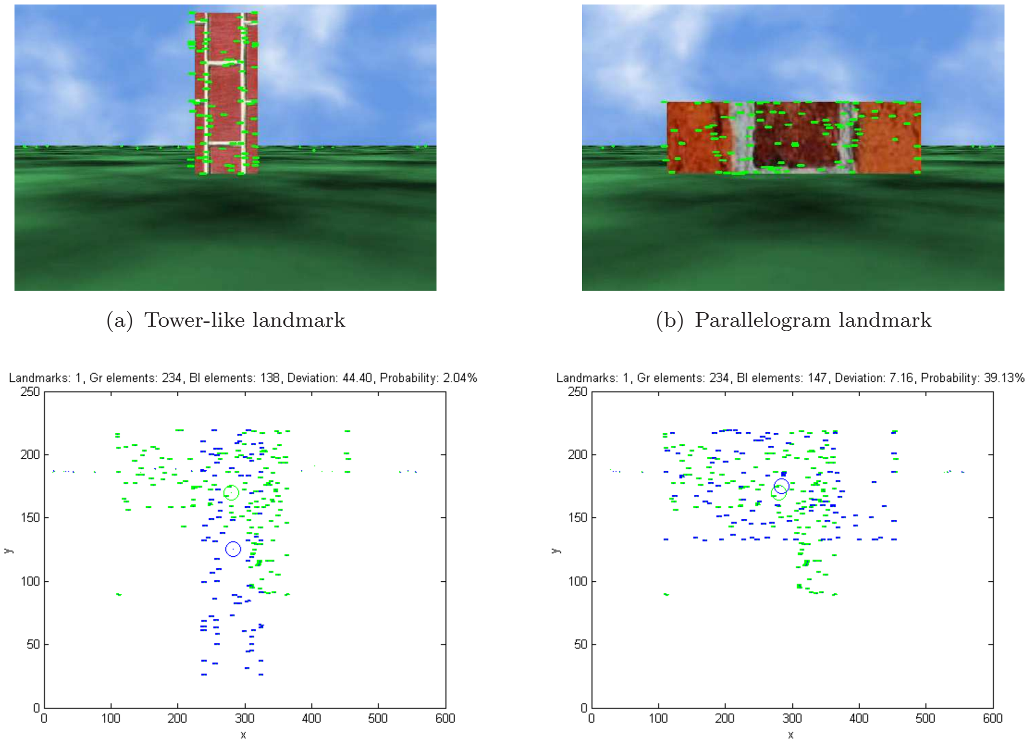
similarity score is high enough, that is 39.13% although the two landmarks are different to each other. In the former case, the probability is low because deviation is large while in the latter, the probability is high because deviation is small. As it can be seen in the last graph, Fig. C.6(d), the texture and the shape of the landmark, Fig. C.6(b), resemble that of the reference landmark, Fig. C.3. In cases such as this, the matching algorithm, and hence the localisation of the robot can be erroneous.

Finally, in Fig. C.7, a comparison between two landmarks is attempted. In the first graph, Fig. C.7(c), two landmarks are captured initially at a distance of $5m$ from the sphere-like landmark and $11m$ from the tower-like landmark, Fig. C.7(a). In the same graph, a second snapshot has been captured but at a distance of $13m$ from the tower-like landmark and $7m$ from the sphere-like landmark. The velocity at which the initial snapshot was taken is $4km/h$ while at the current (second) snapshot the velocity was increased to $5km/h$. Their similarity score is at an acceptable level, that is, 22.55% for the tower-like landmark and 27.08% for the sphere-like landmark. In the graph of Fig. C.7(d) the landmarks of Fig. C.7(a) at their initial snapshot are compared with two different landmarks, Fig. C.7(b). The hill-like landmark is at a distance of $7.0m$ while the tower-like landmark is at a distance of $13m$. The velocity the image was taken is $5km/h$. The probability in this case is quite low revealing the dissimilarity between the landmarks. In particular, the similarity score of the tower-like landmarks is at 8.34% while for the other two landmarks is at 0.93%.

## C.5 Conclusions and Future Work

The simulation experiments of this work show that a similarity score of 20.0% and above is adequate to identify and recognise a landmark from its optical flow 'fingerprint' . The results are quite encouraging and sensible, especially if we take into consideration the fact that the only information used was optic flow. Our method promises to tackle the homing problem in *a priori* unknown environment using a parsimonious biologically-inspired approach to solve a well-studied problem. Of significant interest is that our model can also help explain the methods employed by insects, and in particular honeybees, to perform localisation and thus homing. To support this, a recent study by Avargues-Weber et al. (2009) reveals that honeybees are capable of discriminating faces. It could well be the case of optical flow patterns. In addition, our model does not require the storage or the processing of images every time matching is to be performed. Only the properties of the vectors are stored in every frame, that is, the mean position of all the vectors and the number of vectors.

Future work will focus on using Bayesian statistics to extend the optic flow model to complex scene environments. In this case, a prior distribution needs to be calculated and a likelihood to be determined. The posterior distribution will enable the model to adapt and improve as new observations enter the model. In addition, a robust clustering algorithm for classifying various landmarks would be advantageous. However, the problem in this case is that we do not know which vectors belong to which landmarks. An unsupervised clustering method, therefore, needs to be developed. This problem can also be overcome if there is no counting of landmarks and the images are not fragmented into landmarks, but instead are taken as whole piece of information.



(a) Tower-like landmark

(b) Parallelogram landmark

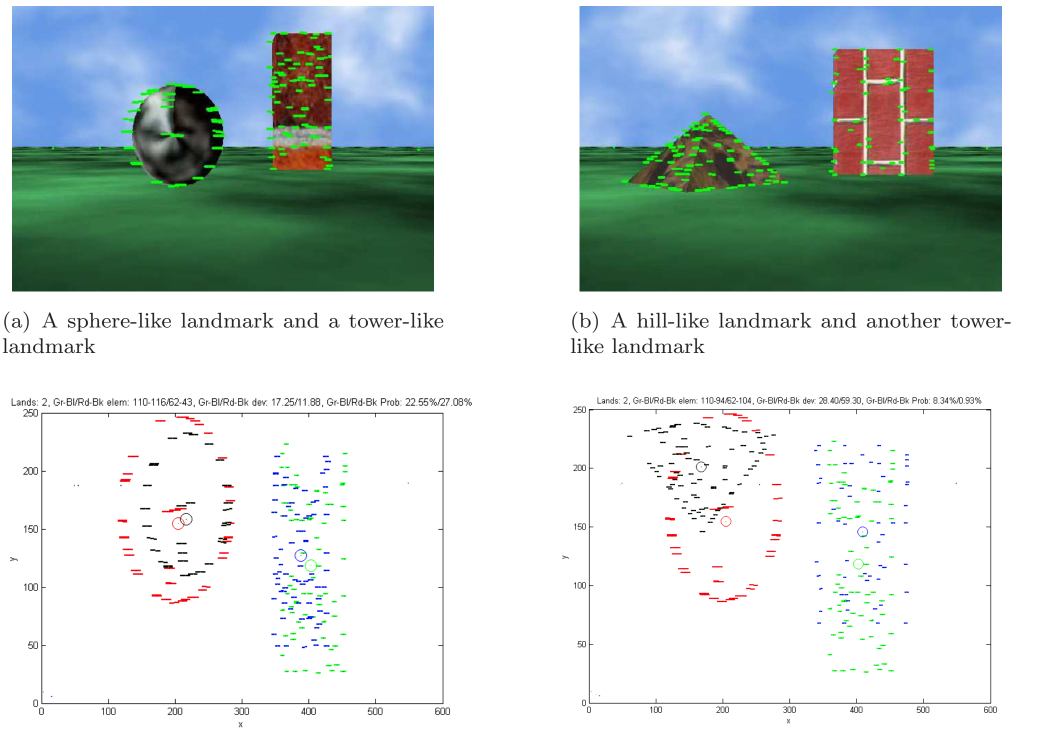
(c) Optic flow vectors between the tower-like landmark and the reference landmark.

(d) Optic flow vectors between the parallelogram landmark and the reference landmark.

Figure C.6: Comparison of optic flow vectors between two different landmarks and the reference landmark. The distance of $11m$ and the velocity of $4km/h$ remains the same as is in the reference landmark.

## Acknowledgment

The authors would like to thank Dr Klaus-Peter Zauner for his guidance and help in shaping the optic flow idea.



(a) A sphere-like landmark and a tower-like landmark

(b) A hill-like landmark and another tower-like landmark

(c) Optic flow vectors captured initially at a distance of $5m$ from sphere-like landmark and $11m$ from the tower-like landmark. The initial velocity at which image was taken is $4km/h$. The current (second) snapshot has been taken at $2m$ further away from the initial snapshot and at a velocity of $5km/h$.

(d) Optic flow vectors between the landmarks of Fig. C.7(a) at their initial snapshot and two new different landmarks, Fig. C.7(b), located at a distance of $11m$ from the tower and $8.0m$ from the hill-like landmark. The velocity the image was taken is $3km/h$.

Figure C.7: Comparison of optic flow vectors between identical and different landmarks.

# References

F. Amigoni, S. Gasparini, and M. Gini. Map building without odometry information. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3753–3758, 2004.

D. An and H. Wang. VPH: A new laser radar based obstacle avoidance method for intelligent mobile robots. In *Proceedings of the Fifth World Congress on Intelligent Control and Automation (WCICA)*, volume 5, pages 4681–4685, 2004.

A. Anderson. A model for landmark learning in the honey-bee. *Journal of Comparative Physiology A*, 114(3):335–355, 1977.

The Grey Walter Online Archive. http://www.ias.uwe.ac.uk/robots/gwonline/gwonline.html, 2006.

A. A. Argyros, C. Bekris, S. C. Orphanoudakis, and L. E. Kavraki. Robot homing by exploiting panoramic vision. *Journal of Autonomous Robots*, 19(1):7–25, 2005.

L. Armesto and J. Tornero. SLAM based on Kalman filter for multi-rate fusion of laser and encoder measurements. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1860–1865, 2004.

A. Avargues-Weber, G. Portelli, J. Benard, A. Dyer, and M. Giurfa. Configural processing enables discrimination and categorization of face-like stimuli in honeybees. *Journal of Experimental Biology*, 213(4):593–601, 2009.

Alireza Bab-Hadiashar and David Suter. Robust total least squares based optic flow computation. In *Proceedings of the 3rd Asian Conference on Computer Vision*, volume 1351, pages 566–573, 1997.

J. Banks and P. Corke. Quantitative evaluation of matching methods and validity measures for stereo vision. *The International Journal of Robotics Research*, 20(7): 512–532, 2001.

J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

S. Benhimane and E. Malis. Homography-based 2D visual tracking and servoing. *The International Journal of Robotics Research*, 26(7):661–676, 2007.

G. Bianco and R. Cassinis. Biologically-inspired visual landmark learning for mobile robots. In *EWLR-8: Proceedings of the 8th European Workshop on Learning Robots*, pages 138–164, London, UK, 2000. Springer-Verlag. ISBN 3-540-41162-3.

M. Blackburn and H. Nguyen. Vision based autonomous robot navigation: motion segmentation. In *Proceedings for the Dedicated Conference on Robotics, Motion, and Machine Vision in the Automotive Industries*, pages 353–360, Stuttgart, Germany, September 1995.

D. Boley, E. Steinmetz, and K. T. Sutherland. Recursive total least squares: An alternative to using the discrete kalman filter in robot navigation. In *Lecture Notes in Artificial Intelligence*, volume 25, pages 221–234, 1995.

G. Bradski and A. Kaehler. *Learning OpenCV, computer vision with the opencv library.* O'Reilly Media, Inc., Sebastopol, CA, USA, 2008.

V. Braitenberg. *Vehicles: experiments in synthetic psychology.* The MIT Press, Cambridge, MA, USA, 1984. ISBN 0-262-52112-1.

R. Brooks. A robust layered control system for a mobile robot. Technical report, Massachusetts Institute of technology, Cambridge, MA, USA, 1985.

R. Brooks. Intelligence without reason. Technical report, Massachusetts Institute of technology, Cambridge, MA, USA, 1991.

T. Camus, D. Coombs, M. Herman, and T-S. Hong. Real-time single-workstation obstacle avoidance using only wide-field flow divergence. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 3, 1996.

B. Cartwright and T. S. Collett. Landmark maps for honeybees. *Biological Cybernetics*, 57(1–2):85–93, 1987a.

B. A. Cartwright and T. S. Collett. Landmark learning in bees. *Journal of Comparative Physiology A*, 151(4):521–543, 1983.

B. A. Cartwright and T. S. Collett. Landmark maps for honeybees. *Biological Cybernetics*, 57(1–2):85–93, 1987b.

K. Cheng, T. S. Collett, A. Pickhard, and R. Wehner. The use of visual landmarks by honeybees: Bees weight landmarks according to their distance from the goal. *Journal of Comparative Physiology A*, 161(3):469–475, 1987.

L. Chittka, J. Spaethe, A. Schmidt, and A. Hickelsberger. Adaptation, constraint, and chance in the evolution of flower color and pollination color vision. In L. Chittka and J. D. Thomson, editors, *Cognitive Ecology of Pollination*, pages 106–126. Cambridge University Press, 2001.

L. Chittka and J. Tautz. The spectral input to honeybee visual odometry. *The Journal of Experimental Biology*, 206(14):2393–2397, 2003.

M. Choi, R. Sakthivel, and W. K. Chung. Neural network-aided extended Kalman filter for SLAM problem. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1686–1690, 2007.

M. Collett, T. Collett, and M. Srinivasan. Insect navigation: Measuring travel distance across ground and through air. *Current Biology*, 16(20):887–890, 2006.

M. Collett, T. S. Collett, and R. Wehner. Calibration of vector navigation in desert ants. *Current Biology*, 9(18):1031–1034, 1999.

T. Collett. Insect vision: Controlling actions through optic flow. *Current Biology*, 12 (18):615–617, 2002.

T. S. Collett. Insect navigation *en route* to the goal: multiple strategies for the use of landmarks. *The Journal of Experimental Biology*, 199(1):227–235, 1996.

T. S. Collett and J. Baron. Biological compasses and the coordinate frame of landmark memories in honeybees. *Nature*, 368(6467):137–140, 1994.

T.S. Collett and M. Collett. Memory use in insect visual navigation. *Nature Reviews Neuroscience*, 3:542–552, 2002.

Michael Csorba. *Simultaneous Localisation and Map Building.* PhD thesis, Department of Engineering Science, University of Oxford, 1997.

M. Dacke and M. Srinivasan. Evidence for counting in insects. *Animal Cognition*, 11 (4):683–689, 2008.

A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6): 1052–1067, 2007.

A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the Ninth International Conference on Computer Vision ICCV '03, Nice, France*, 2003.

A. J. Davison and D. W. Murray. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (7):865–880, 2002.

S. Ch. Diamantas and R. M. Crowder. Localisation and mapping using a laser range finder: A goal-seeking approach. In *Proceedings of the Fifth International Conference on Autonomic and Autonomous Systems*, pages 270–276, Valencia, Spain, April 2009.

S. Ch. Diamantas, A. Oikonomidis, and R. M. Crowder. Depth estimation for autonomous robot navigation: A comparative approach. In *Proceedings of the International Conference on Imaging Systems and Techniques*, Thessaloniki, Greece, July 2010, In Press,a.

S. Ch. Diamantas, A. Oikonomidis, and R. M. Crowder. Towards optical flow-based robotic homing. In *Proceedings of the International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, Barcelona, Spain, July 2010, In Press,b.

G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (slam) problem. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1009–1014, 2000.

G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *IEEE Robotics and Automation Magazine*, 13(2):99–110, 2006.

T. Einsele. Real-time self-localization in unknown indoor environments using a panorama laser range finder. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 697–702, 1997.

H. Esch and J. Burns. Distance estimation by foraging honeybees. *Journal of Experimental Biology*, 199(1):155–162, 1996.

H. Esch, S. Zhang, M. Srinivasan, and J. Tautz. Honeybee dances communicate distances measured by optic flow. *Nature*, 411(6837):581–583, 2001.

D. Fox, S. Thrun, W. Burgard, and F. Dellaert. Particle filters for mobile robot navigation. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo methods in practice*. Springer, 2001.

M. Franz, B. Schölkopf, H. Mallot, and H. Bülthoff. Learning view graphs for robot navigation. *Autonomous Robots*, 5(1):111–125, 1998a.

M. Franz, B. Schölkopf, H. Mallot, and H. Bülthoff. Where did i take that snapshot? scene-based homing by image matching. *Biological Cybernetics*, 79(3):191–202, 1998b.

M. O. Franz and H. A. Mallot. Biomimetic robot navigation. *Robotics and Autonomous Systems*, 30(1–2):133–153, 2000.

H. Frenz and M. Lappe. Absolute travel distance from optic flow. *Vision Research*, 45 (13):1679–1692, 2005.

S. N. Fry and R. Wehner. Look and turn: landmark-based goal navigation in honey bees. *Journal of Experimental Biology*, 208:3945–3955, 2005.

Gazebo. http://playerstage.sourceforge.net/gazebo/gazebo.html, 2006.

GDAL. http://www.gdal.org/, 2006.

B. P. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: tools for multi-robot and distributed sensor systems. In *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)*, pages 317–323, Coimbra, Portugal, June 30 - July 3 2003.

J. J. Gibson. *The Perception of the Visual World*. Greenwood Publishing Group, Santa Barbara, CA, USA, 1974.

G. Grah, R. Wehner, and B. Ronacher. Path integration in a three-dimensional maze: ground distance estimation keeps desert ant Cataglyphis fortis on course. *Journal of Experimental Biology*, 208(21):4005–4011, 2005.

P. Graham and T. S. Collett. View-based navigation in insects: how wood ants (*Formica rufa* l.) look at and are guided by extended landmarks. *Journal of Experimental Biology*, 205(16):2499–2509, 2002.

V. V. Hafner. *Adaptive navigation strategies in biorobotics: visual homing and cognitive mapping in animals and machines*. Shaker Verlag GmbH, Aachen, 2004. ISBN 3-8322-2857-8.

D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose. Mapping and localization with RFID technology. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 1015–1020, 2004.

D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser measurements. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 206–211, 2003.

C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

J. F. Harrison, J. H. Fewell, T. M. Stiller, and M. D. Breed. Effects of experience on use of orientation cues in the giant tropical ant. *Animal Behavior*, 37(5):869–871, 1989.

S. Healy. *Spatial representation in animals*. Oxford University Press, Oxford, UK, 1998.

O. Holland. Exploration and high adventure: the legacy of Grey Walter. *Philosophical Transactions of the Royal Society*, 361(1811):2085–2121, 2003.

J. Hong, X. Tan, B. Pinette, R. Weiss, and E. M. Riseman. Image-based homing. In *IEEE International Conference on Robotics and Automation*, pages 620–625, 1991.

B. K. P. Horn and G. Schunck. Determining optical flow. *Artificial Intelligence*, 17: 185–203, 1981.

S. Hrabar, G. Sukhatme, P. Corke, K. Usher, and J. Roberts. Combined optic-flow and stereo-based navigation of urban canyons for a uav. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 302–309, August 2005.

Chun jen Tsai, Nikolas P. Galatsanos, and Aggelos K. Katsaggelos. Total least squares estimation of stereo optical flow. In *Proceedings of the IEEE International Conference on Image Processing*, pages 622–626, 1998.

Chun jen Tsai, Nikolas P. Galatsanos, and Aggelos K. Katsaggelos. Optical flow estimation from noisy data using differential techniques. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3393–3396, 1999.

M. Jogan and A. Leonardis. Robust localization using panoramic view-based recognition. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 4, pages 136–139, 2000.

S. P. D. Judd and T. S. Collett. Multiple stored views and landmark guidance in ants. *Nature*, 392(6677):710–714, 1998.

F. Kendoul, I. Fantoni, and K. Nonami. Optic flow-based vision system for autonomous 3d localization and control of small aerial vehicles. *Robotics and Autonomous Systems*, 57(6–7):591–602, 2009.

J. Klein. Breve: a 3d simulation environment for the simulation of decentralized systems and artificial life. In *Proceedings of Artificial Life VIII, the 8th International Conference on the Simulation and Synthesis of Living Systems*, 2002.

J. H. Klotz. Topographic orientation in two species of ants (hymenoptera: Formicidae). *Insectes Sociaux*, 34(4):236–251, 1987.

N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2149–2154, Sendai, Japan, September 28 - October 2 2004.

Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1398–1404, 1991.

K. Kral and M. Poteser. Motion parallax as a source of distance information in locusts and mantids. *Journal of Insect Behavior*, 10(1):145–163, 1997.

D. Lambrinos, M. Maris, H. Kobayashi, T. Labhart, R. Pfeifer, and R. Wehner. An autonomous agent navigating with a polarized light compass. *Adaptive Behavior*, 6 (1):131–161, 1997.

D. Lambrinos, R. Moller, T. Labhart, R. Pfeifer, and R. Wehner. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems, special issue: Biomimetic Robots*, 30(1–2):39–64, 2000.

M. Langer and R. Mann. Optical snow. *International Journal of Computer Vision*, 55 (1):55–71, 2003.

M. Lehrer. Bees which turn back and look. *Naturwissenschaften*, 78(6):274–276, 1991.

M. Lehrer. Why do bees turn back and look? *Journal of Comparative Physiology A*, 172(5):549–563, 1993.

M. Lehrer and G. Bianco. The turn-back-and-look behaviour: bee versus robot. *Biological Cybernetics*, 83(3):211–229, 2000.

M. Lehrer and T. S. Collett. Approaching and departing bees learn different cues to the distance of a landmark. *Journal of Comparative Physiology A*, 175(2):171–177, 1994.

J. Leonard, H. F. Durrant-Whyte, and I. Cox. Dynamic map building for an autonomous mobile robot. *The International Journal of Robotics Research*, 11(4):286–298, 1992.

D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision (ICCV '99)*, pages 1150–1157, Corfu, Greece, September 1999.

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

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI), August 24-28*, pages 674–679, 1981.

H. Madjidi and S. Negahdaripour. On robustness and localization accuracy of optical flow computation for underwater color images. *Computer Vision and Image Understanding*, 104(1):61–76, 2006.

Aamir Saeed Malik and Tae-Sun Choi. Depth estimation by finding best focused points using line fitting. In *Proceedings of the 3rd International Conference on Image and Signal Processing*, volume 5099, pages 120–127, 2009.

J. L. Martinez, A. Pozo-Ruz, S. Pedraza, and R. Fernandez. Object following and obstacle avoidance using a laser scanner in the outdoor mobile robot Auriga-$\alpha$. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 204–209, October 13-17 1998.

MATLAB. http://www.mathworks.com/, 2005.

F. Matsuno and S. Tadokoro. Rescue robots and systems in Japan. *IEEE International Conference on Robotics and Biomimetics*, pages 12–20, August 2004.

S. J. Maybank. Algorithm for analysing optical flow based on the least-squares method. *Image Vision Computing*, 4(1):38–42, 1986.

R. Menzel, K. Geiger, J. Joerges, and U. Muller L. Chittka. Bees travel novel homeward routes by integrating separately acquired vector memories. *Animal Behaviour*, 55(1): 139–152, 1998.

P. C. Merrell, D-J. Lee, and R. Beard. Obstacle avoidance for unmanned air vehicles using optical flow probability distributions. *Sensing and Perception*, 5609:13–22, 2004.

R. Möller. Insect visual homing strategies in a robot with analog processing. *Biological Cybernetics, special issue: Navigation in Biological and Artificial Systems*, 83(3):231–243, 2000.

R. Moller. Do insects use templates or parameters for landmark navigation? *Journal of Theoretical Biology*, 210(1):33–44, 2001.

R. Möller, D. Lambrinos, P. Pfeifer, T. Labhart, and R. Wehner. Modeling ant navigation with an autonomous agent. In *From Animals to Animats 5: Proceedings of the 5th International Conference on Simulation of Adaptive Behavior*, pages 185–194, 1998.

M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002.

T. Mori, Y. Matsumoto, T. Shibata, M. Inaba, and H. Inoue. Trackable attention point generate based on classification of correlation value distribution. In *JSME Annual Conference on Robotics and Mechatronics (ROBOMEC '95)*, pages 1076–1079, 1995.

M. Müller and R. Wehner. The hidden spiral: systematic search and path integration in desert ants, *Cataglyphis fortis. Journal of Comparative Physiology A*, 175(5):525–530, 1994.

R. C. Nelson. Visual homing using an associative memory. *Biological Cybernetics*, 65 (4):281–291, 1991.

P. Newman, J. Leonard, J. D. Tardos, and J. Neira. Explore and return: Experimental validation of real time concurrent mapping and localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1802–1809, May 2002.

ODE. http://www.ode.org/, 2006.

N. Ohnishi and A. Imiya. Corridor navigation and obstacle avoidance using visual potential for mobile robot. In *Proceedings of the Fourth Canadian Conference on Computer and Robot Vision*, pages 131–138, May 2007.

OpenCV. http://opencv.willowgarage.com/wiki/, 2008.

OpenGL. http://www.opengl.org/, 2006.

Player. http://playerstage.sourceforge.net/player/player.html, 2006.

Player/Stage. http://playerstage.sourceforge.net/, 2006.

S. Pratt, S. Brooks, and N. Franks. The use of edges in visual navigation by the ant *leptothorax albipennis*. *Ethology*, 107(12):1125–1136, 2001.

A. D. Redish. *Beyond the cognitive map: from place cells to episodic memory.* The MIT Press, Cambridge, MA, USA, 1999.

Mobile Robots. http://www.activrobots.com/robots/systems.html, 2006.

B. Ronacher, K. Gallizzi, S. Wohlgemuth, and R. Wehner. Lateral optic flow does not influence distance estimation in the desert ant Cataglyphis fortis. *Journal of Experimental Biology*, 203(7):1113–1121, 2000.

Ishikawa Satoru, Yoneda Masaaki, Hase Hiroyuki, and Sakai Mitsuru. Depth estimation by optical flow method from images including moving objects. *Journal of the Institute of Image Electronics Engineers of Japan*, 28(1):48–55, 1999.

S. Schuster, R. Strauss, and K. G. Gotz. Virtual-reality techniques resolve the visual cues used by fruit flies to evaluate object distances. *Current Biology*, 12(18):1591–1594, 2002.

N. Sharkey. Biologically inspired robotics. *The Handbook of Brain Theory and Neural Networks*, pages 160–164, 2002.

J. Shi and C. Tomasi. Good features to track. In *Proceedings of the 9th IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

W. Simpson. Optic flow and depth perception. *Spatial Vision*, 7(1):35–75, 1993.

Natalia Slesareva, Andrés Bruhn, and Joachim Weickert. Optic flow goes stereo: A variational method for estimating discontinuity-preserving dense disparity maps. In *27th DAGM Symposium*, volume 3663, pages 33–40, 2005.

R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *Fourth International Symposium on Robotics Research*, pages 467–474, 1988.

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

M. Srinivasan, J. Chahl, K. Weber, S. Venkatesh, M. Nagle, and S. Zhang. Robot navigation inspired by principles of insect vision. *Robotics and Autonomous Systems*, 26(2):203–216, 1999.

M. V. Srinivasan, S. Zhang, M. Altwein, and J. Tautz. Honeybee navigation: Nature and calibration of the "odometer". *Science*, 287(5454):851–853, 2000.

M. V. Srinivasan, S. W. Zhang, M. Lehrer, and T. S. Collett. Honeybee navigation *en route* to the goal: Visual flight control and odometry. *Journal of Experimental Biology*, 199(1):237–244, 1996.

Stage. http://www-robotics.usc.edu/interaction/teaching/psg.pdf, 2006.

W. Stürzl and H. Mallot. Vision-based homing with a panoramic stereo sensor. In *Biologically Motivated Computer Vision: Second International Workshop, BMCV 2002, Tübingen, Germany, November 22-24*, volume 2525, pages 620–628, 2002.

R. M. Sunderland. *Biologically Inspired Control Techniques for Compliant Reaching.* PhD thesis, School of Electronics and Computer Science, University of Southampton, 2006.

J. Tautz, S. Zhang, J. Spaethe, A. Brockmann, A. Si, and M. Srinivasan. Honeybee odometry: Performance in varying natural terrain. *Plos Biology*, 2(7):915–923, 2004.

S. Thompson, A. Zelinsky, and M. Srinivasan. Automatic landmark selection for navigation with panoramic vision. In *Proceedings of Australian Conference on Robotics and Automation (ACRA '99)*, Brisbane, Australia, March 1999.

S. Thrun, D. Fox, and W. Burgard. Monte carlo localization with mixture proposal distribution. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2000.

S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Journal of Artificial Intelligence*, 128(1-2):99–141, 2001.

O. Trullier, S. Wiener, A. Berthoz, and J. Meyer. Biologically-based artificial navigation systems: review and prospects. *Progress in Neurobiology*, 51:483–544, 1997.

I. Ulrich and J. Borenstein. VFH+: Reliable obstacle avoidance for fast mobile robots. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 1572–1577, Leuven, Belgium, May 16-21 1998.

A. Vardy. *Biologically Plausible Methods for Robot Visual Homing.* PhD thesis, School of Computer Science, Carleton University, 2005.

K. von Frisch. *The Dance Language and Orientation of Bees.* Harvard University Press, Cambridge, MA, USA, 1993.

W. Warren and B. R. Fajen. From optic flow to laws of control. In L. M. Vaina, S. A. Beardsley, and S. K. Rushton, editors, *Optic Flow and Beyond*, pages 307–337. Kluwer Academic Publishers, 2004.

B. Webb. Can robots make good models of biological behaviour? *Behavioral and Brain Sciences*, 24(6):1033–1050, 2001.

R. Wehner. Polarization vision - a uniform sensory capacity. *Journal of Experimental Biology*, 204(14):2589–2596, 2001.

R. Wehner, B. Michel, and P. Antonsen. Visual navigation in insects: Coupling of egocentric and geocentric information. *Journal of Experimental Biology*, 199(1):129–140, 1996.

R. Wehner and F. Raber. Visual spatial memory in desert ants, Cataglyphis bicolor. *Experientia*, 35(12):1569–1571, 1979.

R. Wehner and M. Srinivasan. Searching behaviour of desert ants, genus *Cataglyphis* (formicidae, hymenoptera). *Journal of Comparative Physiology A*, 142(3):315–338, 1981.

G. Welch and G. Bishop. An introduction to the kalman filter. Technical Report TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 2006.

N. Wiener. *Cybernetics: or the control and communication in the animal and the machine.* The MIT Press, Cambridge, MA, USA, 1965.

M. Wittlinger, R. Wehner, and H. Wolf. The desert ant odometer: a stride integrator that accounts for stride length and walking speed. *The Journal of Experimental Biology*, 210(2):198–207, 2007.

S. Zhang and M. V. Srinivasan. Behavioural evidence for parallel information processing in the visual system of insects. *The japanese Journal of Physiology*, 43(1):247–258, 1993.

J-C. Zufferey, A. Beyeler, and D. Floreano. Optic flow to control small uavs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems: Workshop on Visual Guidance Systems for small autonomous aerial vehicles*, Nice, France, September 2008.