2013-01-18

Goal-driven Mobile Robot Navigation in Unknown Indoor Environments

Marasinghe Arachchige, Nuwan Ganganath

UNIVERSITY OF CALGARY

Goal-driven Mobile Robot Navigation in Unknown Indoor Environments

by

Nuwan Ganganath Marasinghe Arachchige

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

January, 2013

Abstract

A goal-driven mobile robot navigation system is proposed for unknown indoor environments. The proposed system can be subdivided into four main modules, namely, localization, mapping, motion control, and goal detection.

In the proposed system, a mobile robot uses an odometry system and a Kinect sensor as its input devices. An optimal particle filter models the posterior over the robot trajectory while minimizing the variance of the importance weights of the particles. Occupancy grid maps are employed to represent the environment as they do not make any assumptions on distinguishable landmarks. A nearness-diagram reactive navigation technique generates motion commands based on the robot position and navigation goal location. A trajectory parameter space is used as an abstraction layer of the robot shape and kinematic constraints for the nearness-diagram method. A goal-driven situation assessment framework based on fuzzy cognitive maps is developed to verify navigation goals using sensory information and expert knowledge.

Acknowledgements

I owe my deepest gratitude to my supervisor, Dr. Henry Leung for all the suggestions, advice, guidance, and encouragement given to me throughout the course of my masters research. Also, I respectfully admire his thirst for new knowledge, search for real-world applications for signal processing, devotion as a teacher, and his generosity.

I thank my teachers at the University of Calgary including Dr. John Nielsen, Dr. Raj Rangayyan, Dr. Michel Fattouche, and Dr. Alejandro Ramirez-Serrano for all their support. I extend my appreciation to Dr. Chi-Tsun Cheng at the Hong Kong Polytechnic University, Hong Kong, Dr. Ranga Rodrigo and Dr. Ajith Pasqual at the University of Moratuwa, Sri Lanka, and Dr. Arjuna Madanayake at the University of Akron, USA for their guidance.

I am also grateful to my friends and colleagues, Amila Silva, Thushara Gunaratne, Achintha Maddumabandara, Udara Waraketiya, Joydeep Mukherjee, Sumona Mukhopadhyay, Chatura Wickrema Seneviratne, Martin Walker, Pristley Sathyaraj, and others for their wonderful friendship.

Most importantly I would like to thank my parents for their sempiternal support and love.

This thesis was made possible by a Graduate Teaching/Research Assistantship at the University of Calgary.

Dedicated to my parents,

Piyadasa Marasinghe and Soma Edirisinghe.

Table of Contents

List of Tables

List of Figures and Illustrations

List of Abbreviations

Symbol	Definition
C-Space	Configuration Space
CPR	Counts per Revolution
EKF	Extended Kalman Filter
ESS	Effective Sample Size
FCM	Fuzzy Cognitive Map
FOV	Field-of-view
FPG	Fuzzy Probability Generator
GPS	Global Positioning System
HSV	Hue, Saturation, and Value
IR	Infrared
MCL	Monte Carlo Localization
NASA	National Aeronautics and Space Administration
NBV	Next-best-view
ND	Nearness Diagram
NN	Nearest Neighbor
PCL	Point Cloud Library
PF	Particle Filter
RBPF	Rao-Blackwellied Particle Filter
RGB	Red, Green, and Blue
RMSE	Root Mean Square Error
SAF	Situation Assessment Framework
SAR	Search and Rescue
SLAM	Simultaneous Localization and Mapping

TP-Space	Trajectory Parameter Space
USB	Universal Serial Bus
VFH	Viewpoint Feature Histogram
WMR	Wheeled Mobile Robot

Chapter 1

Introduction

Robotics is an interdisciplinary field of study which combines the technologies from mechanical engineering, electrical engineering, biological science, and computer science. Active research in robotics has revolutionized many aspects of our lives. It finds useful applications in various fields such as industrial assembly lines [1, 2], household environments [3], surveillance [4], search and rescue [5], military battle fields [6, 7], automotive industry [8], medical domain [9], and planetary explorations [10, 11]. Such robots have to fulfill highly demanding goals while coping with external constraints. Therefore most of the current robots are inspirited by nature [12], especially by human characteristics and behaviors [13]. With the present development in robotics, robots have become more efficient compared to their biological counterparts in many applications [14, 15]. They can now replace humans in many dangerous and hazardous environments.

Traditional robotics mainly consists of *manipulator arms* which can be considered as a combination of several links and joints. Many of those manipulator arms imitate human arm in both appearance and functionality. They have been successfully utilized in assembly lines in industrial manufacturing factories all over the world. An end effector of a manipulator arm, the counterpart of human hand, reaches a point of interest in its workspace by using transactional and rotational motions between its links. However, the traditional manipulator arms are generally attached to a fixed point in the environment which limits them to stationary workspaces. Therefore, the lack of mobility is an obvious disadvantage of traditional manipulator arms.

In contrast to traditional robots, *mobile robots* are capable of moving in a given environment without being attached to a fixed location. They use different locomotion

mechanisms to move themselves from place to place. Types of locomotion heavily depend on the environment and applications that they are utilized in. With the added advantage of extra mobility over traditional robot arms, mobile robots become useful in many real world applications. In most of the situations, they are capable of reaching the exact point where their services are required. Motion control of mobile robots can be either manual or autonomous. Manual mobile robots are normally teleoperated by a human using their own vision and knowledge of the environment or using perception aids, such as cameras. These can be a single camera attached to the robot itself or multiple cameras placed in its workspace. In contrast to teleoperated mobile robots, autonomous mobile robots have to make their own decisions depending on their perception of the external world.

Autonomous mobile robotics has become an extremely challenging problem due to inherent unpredictability of the physical world, limitations of real-time information, and incomplete prior knowledge. The level of unpredictability of the environment is governed by two main factors: a prior knowledge and randomness of the environment. The mobile robots which work in well structured environments such as industrial assembly lines or office rooms can use their prior knowledge about this environment to complete their tasks, as the structural change of environment over time is minimal. Nevertheless, in most of the situations the robots have to deal with highly dynamic, unstructured, very dense, and complex environments. Even a well structured environment can be highly unpredictable when people are working in there. This can be easily understood by identifying the difference of a cafeteria in a holiday and during busy hours. The situation becomes even worse if the robot does not have any prior knowledge about its target environment.

Imagine you are suddenly dropped into a totally unknown place, say a city center, and asked to go to a coffee shop with a specific name. Without any external support

such as maps, how could you achieve this? You may use your vision system to see the environment around you. You may also try to extract key features which will eventually help you to understand the nature of the environment. You may localize yourself compared to other objects in the environment. Somehow you do not know where you are globally and you do not need to know it as your objective is to find a certain coffee shop available in your local environment. After you have localized yourself in the given environment, what will be the next step? You need to move in a direction where you can find the coffee shop. Without any prior knowledge about the current environment, this can be any random direction. However you may still be able to make an intelligent guess with similar kind of environments you have previously been in. Finally, you may start walking while further learning your environment, *i.e.* now you have more knowledge. How to identify your goal, viz. the coffee shop, while moving? Is it possible to do it just by reading names? Similar words might be available in advertising posters put on some other places in the city. Therefore you need to analyze and understand the situation to achieve your target successfully. Dealing with physical environments are challenging for the so called most intelligent creation of the natural world, the humans, as it is for the robots. In many real-world applications, the mobile robots have to deal with such challenging environments with limited computational resources.

This thesis addresses the problems in goal-driven mobile robot navigation in unknown environments. Section 1.1 of this chapter identifies the challenges in autonomous mobile robotics, especially in unknown environments. Existing solutions and their drawbacks are reviewed in Section 1.2. The organization of this thesis is briefly described in Section 1.3.

1.1 Challenges for Mobile Robots in Unknown Environments

With rapid advancements in science and technology, scientists' curiosity has stepped out of the planet earth and reached up to a level of exploring moon and other planets in the solar system. The achievements of the National Aeronautics and Space Administration (NASA) agency of the United States can be taken as examples for that. NASA's Curiosity rover captured considerable public attention in the recent past for its exploration missions in planet Mars [16]. Terrestrial planet exploration rovers have to learn their environment while navigating around. The main objective of such robots is to maximize the knowledge about their environment within the shortest possible time. However, a pertinent and challenging issue remains unsolved to a large extent in finding solutions which deal with such situations where prior knowledge about target environment is unknown. In the rest of this section, search and rescue (SAR) robots are taken as an example to specifically identify the challenges for autonomous mobile robots that are employed in previously unknown and highly demanding environments.

Hazardous situations arise mainly because of the irresistible natural causes such as earthquakes, tsunamis, tornados, and forest fire. Somehow, similar or even more dangerous situations might occur due to man-made disasters such as the collapse of the World Trade Center in New York City in 2001 and the Fukushima Daiichi nuclear disaster in 2011. Either natural or man-made, rescue missions in these kind of situations are dangerous for human rescuers. If we consider a SAR situation in an urban environment, the rescuers usually have to deal with collapsed buildings, fire and explosions, victims trapped inside buildings, and other inherent problems to such environments. The autonomous mobile robots can be used in these kind of situations to assist rescue teams. The use of autonomous robots are invaluable, specially during an initial phase of a disaster, which is highly dangerous for human rescuers to deal with. Are robots capable of handling such situations?

One of the main challenges of SAR robot missions is the lack of information. Normally the structure of an environment looks totally different after a disaster. Therefore the rescue robots have minimal or no information about the modified environment as they are unable to use the previous maps or other form of knowledge. Hence, they need to sense their environment to acquire required information. *Sensing* is one of the most important tasks of any kind of autonomous mobile robots. The type of the sensors used with autonomous robot are directly dependent upon the application of the robots and the nature of the environment. Therefore one needs to carefully select the sensors for robot platforms. One of the most common sensor types used in the SAR robots is cameras. They are helpful in capturing visual information of the surrounding environment which can be used to recognize the surrounding objects using computer vision techniques. Therefore robots should be able to distinguish people from other man made objects. However, the visual information gathered from single camera is not enough to navigate in the environment without any collisions. Because object avoidance cannot be achieved by simply identifying obstacles, the robots need to measure the distance to them. This can be accomplished either by using a stereo camera pair or by using depth sensors such as laser range finders or sonar sensors. Apart from that, the robots can use Global Positioning System (GPS) to identify their locations.

However, sensors have their own limitations. For example, normal cameras cannot operate in the dark. Infrared cameras can be used to solve the problem, but they are unable to capture all the visual features on a surface. Ultrasonics sensors suffer the limited bandwidth and cross-talk. Laser range finders have successfully overcome these problems. However, laser range finders cannot detect transparent materials such as glass which can be easily detected using ultrasonic sensors. Likewise, each sensor has unique problems. Meanwhile, measurements of all the sensors are subjected to noise which degrades the quality of measurements. *Uncertainty of sensor measurements* is a

major problem which needs be addressed. This problem is usually addressed either by improving the quality of the materials and manufacturing process of these sensors or using advanced algorithms to filter the sensor noise.

Although GPS is useful to localize mobile robots in outdoor environments, it cannot be used in indoor or very dense environments. Unfortunately in many situations robots have to work in such environments. Especially during rescue missions inside buildings, SAR are unable to use GPS to localize themselves. *Localization* is defined as a process of estimating the robot's pose compared to an external reference frame [17]. It is a key task for any kind of mobile robots due to several reasons. Since the perception of a mobile robot depends on its current pose, accurate localization is required to accumulate those sensory information compared to an external reference frame. Mobile robot pose estimation data are also important in motion planning. Therefore it is necessary to use some other localization mechanisms when GPS fails. In some of the indoor mobile robot applications, local positioning systems can be used to localize the robots. However these type of solutions cannot be used to localize rescue robots. Odometry is another popular localization method for the wheeled mobile robots (WMRs), but they suffer from accumulating errors which result in uncertainty in pose estimations.

In SAR missions, robots have to store their knowledge in a way such that both themselves and humans can use it later. As an example, SAR robots might be asked to explore open pathways in a hazardous environment. Also they might have to provide the locations of victims such that rescue teams can reach there within shortest possible time. Therefore, *knowledge representation* is another important task for mobile robotics. This is usually accomplished using robotics maps. Similar to mobile robot localization, *mapping* has attracted a considerable attention of the mobile robot research community. Robotics maps are useful not only for third parties, but also for the robots themselves to plan their own motion. Mapping is a challenging problem for robots due to several

reasons.

If the environment is considerably large compared to the range of the sensors of the mobile robot, the map building might be difficult and time consuming. Also the robot might have to navigate in the environment to capture the whole map. This process definitely increases the uncertainty of the map due to the uncertainty of the localization and actuation. There are no perfect localization and actuation systems that exist in real robots. Noisy sensors and perceptual ambiguity can also add to the uncertainty of the robotics maps. Uncertainty of the map will later result in uncertainty of pose estimation when the robot tries to localize itself in the current map. Therefore, robotics researchers identify this as a *chicken-and-egg* problem and deal with it as the *simultaneous localization and mapping problem* (SLAM) [17]. However, solutions to SLAM are passive, *i.e.* they do not generate motion commands (actions) to guide the robot. *Action selection* is challenging in fully or partially unknown environments. There is a significant impact from action selection on the map built by SLAM. The task of controlling a robot in order to maximize its knowledge about the external environment using its sensors is referred as *exploration* [17]. Therefore, robotic exploration should be achieved through an integrated system which considers SLAM and action selection simultaneously.

The action selection or motion control of mobile robots also depends on its navigation target. However, in unknown environments, it is not possible to locate the target beforehand. Therefore robots need to *understand the environment* and estimate their navigation goals autonomously. If we draw our attention to the previous example again, *i.e.* the mapping of victims in an hazardous environment, how can a rescue robot distinguish victims from normal people or rescuers? It is not possible to use face recognition algorithms or human detection algorithms for that purpose. In order to deal with the ambiguity of highly complex environments, the mobile robots need to employ high level

data fusion techniques to analyze the real-world situations.

1.2 Review of Existing Solutions

The topic of decision making under uncertainly has been well studied in both science and engineering. As pointed out in Section 1.1, the uncertainty of pose estimation of WMRs basically depends on the odometry errors. The most common approach of eliminating the odometry errors is by using an auxiliary sensor to observe landmarks in an environment [18, 19, 20]. Different sensors, such as cameras [21], sonar sensors [22], and laser range finders [22] have been used to detect landmarks and obtain the required measurements. Leonard and Durrant-Whyte employ an *extended Kalman filter* (EKF) for mobile robot localization using geometric beacons which were extracted from sonar scans [23]. Salichs *et al.* also propose an EKF based mobile robot localization system with the aid of artificial landmarks [24]. Using artificial landmarks is both economical and feasible in some indoor environments. Kurazume *et al.* propose localization with multiple robots, where one of them equipped with a sophisticated laser range finder and other robots are used as movable landmarks [25, 26]. *Particle filters* (PFs) for mobile robot localization are first proposed by Dellaert *et al.* [22] and Fox *et al.* [27]. This is also referred as *Monte Carlo localization* (MCL). In robotics literature, many solutions have been proposed for mobile robot localization problem using PFs with cameras [28, 29, 30] while others [30, 31, 32] have proposed combining MCL with omnidirectional cameras. However, these methods assume prior knowledge about the environment which is not always available in many practical scenarios. Therefore, a growing attention is devoted by researchers for robotic exploration problem while minimizing the uncertainty of the environment.

Mobile robot exploration algorithms are mainly based on decision theory and information theory. Most of these techniques focus on acquiring information about the

robot's environment in the shortest possible time. Robotic maps are the most common representation of environmental information that they gathered using those techniques. Koenig *et al.* introduce an early exploration technique for learning topological maps [33]. On the other hand, Thrun introduce the idea of actively exploring for occupancy grid maps using dynamic programming [34]. In that work, real-world knowledge is presented using artificial neural networks and these networks are used to transfer knowledge across different environments once trained. Exploration strategies for feature based maps can be found in [35]. Cassandra *et al.* describe an exploration approach using the idea of information maximization [36]. They have formulated the action selection in mobile robot navigation as a partially observable *Markov decision process*.

In order for the robot to move, it should have a target position. The navigation target selection is one of the main problems in exploration. Yamauchi *et al.* propose their *frontier-based exploration* technique in [37, 38]. Frontier cells define the boundary between explored and unexplored areas. These frontier cells offer the robot a possibility of visiting new places. If no more frontier cells exist in the map, the robot has explored the total area under consideration and the navigation process can be stopped. Gonzalez-Banos *et al.* consider this to be similar to the *next-best-view* (NBV) problem in computer vision [39]. They introduce the concept of a safety region, which is the largest region that is guaranteed to be free of obstacles given the sensor readings made so far. The NBV position is chosen within the safe region in order to maximize the information gain. This approach also proposes how to keep a minimal overlap with the current global map, in order to allow for the registration of successive views under the localization uncertainty of the robot.

The need for multi robot exploration stems from the objective of maximizing the area coverage. A team of collaborative robots has some clear advantages over a single robot: fault tolerance, faster task completion, and compensation of sensor uncertainty.

However, multi-robot exploration techniques have their own problems, such as balancing the spatial distribution of the robots, collision avoidance among themselves, and communicating with other robots. The greedy exploration idea is adapted to teams of collaboratively exploring robots to maximize the map information in [40, 41]. Dias *et al.* propose a robust algorithm for multi robot coordination in dynamic environments [42]. Their technique addresses three existing malfunctions of the multi-robot systems: communication failures, partial failure of robot resources necessary for task execution, and complete robot failure. The minimization of the localization uncertainty in multi-robot systems is discussed in [43]. Burgard *et al.* introduce a decision-theoretic approach to coordinate the robots in order to maximize the overall utility and minimize the potential for overlap in information gain while accomplishing their goal quickly [44]. Brass *et al.* propose a graph based approach where they modeled an obstacle dense environment as a graph which is initially unknown and the existence of the edges become known, as the robots explore the environment [45]. More recently, a distributed value function for multi-robot exploration was introduced which enables each robot to decide upon a local strategy that minimizes the interactions between the robots and maximizes the space coverage of the team [46]. However, this thesis focuses on single-robot exploration techniques. A brief review of the multi robot exploration is presented for the sake of completeness.

Within the context of SLAM, some exploration techniques have been proposed to actively control the robot during SLAM. Makarenko *et al.* propose an approach to integrated exploration [47]. Their algorithm detects the landmarks in laser range data and uses an EKF to solve the SLAM problem. It simultaneously determines the actions to be carried out. The next best action is selected according to the utility function which was designed to favor destinations that offer higher information gain. More accurate localization has been obtained by revisiting the landmarks. Newman *et al.* introduce

a similar feature-based exploration technique in which the next robot action was determined using the geometric, spatial and stochastic characteristics of the current map [48]. The location uncertainty of each feature in the map is represented by a set of probability distribution functions that are used with previous robot actions to determine the next best action. The trajectory planning in SLAM was addressed in [49] by Sim *et al.* Their control algorithm utilizes a parameterized class of spiral trajectory policies with EKF-SLAM to create a map as large as possible. All these exploration techniques are based on the distinguishable features in the environment. These features should be uniquely determined during SLAM.

A widespread category of exploration approaches to SLAM is the use of grid maps which make no assumptions of distinguishable landmarks. Bourgault *et al.* propose an exploration algorithm and demonstrated it in an indoor environment using occupancy grid maps with SLAM [50]. The mapping accuracy is increased by adaptively selecting the control actions during exploration that maximize the localization accuracy. Stachniss and Burgard introduce an integrated technique combining the motion control and grid-based version of the FastSLAM algorithm [51]. Revisiting already explored areas increased the localization accuracy in their approach. In the next step, Stachniss *et al.* introduce the highly efficient Rao-Blackwellized particle filter for active SLAM [52]. Their decision theoretic approach decides possible actions after considering the uncertainty of both map and robot pose. However the RBPF-SLAM in [52] suffers from degeneracy problem which occurs due to the variance of particle weights growing with time. Mobile robot localization, mapping, and exploration algorithms are extensively reviewed in [17].

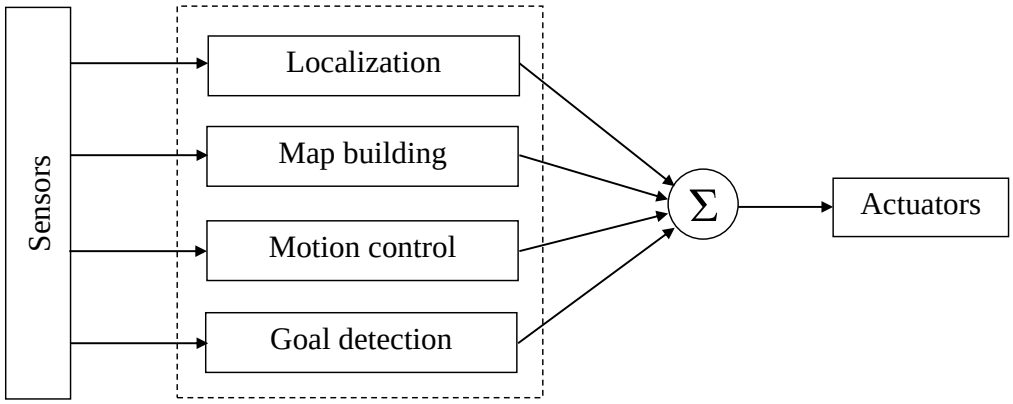


Figure 1.1: Overview of the proposed goal-driven mobile robot navigation system.

1.3 Layout and Contributions of Thesis

In this thesis, the goal-driven mobile robot navigation problem is addressed with an integrated approach as illustrated in Figure 1.1. More detailed descriptions on each component can be found in the rest of the thesis, which is divided into five chapters.

Chapter 2 describes the basics of robot motion and perception in the scope of WMRs. It also includes details on the *mobile robot platform* and *sensors* used in this work. WMR motion models and their applications are also described. An indoor mobile robot *localization* method using an inexpensive sensor system is proposed in Chapter 3. The information fusion of sensory data is achieved with an EKF and a PF in order to minimize the accumulating odometry errors. In Chapter 4, the mobile robot exploration system is proposed by performing *localization*, *map building*, and *motion control* simultaneously. In the proposed integrated approach, an optimal particle filter SLAM is employed which addresses the degeneracy problem. Motion control of the robot is accomplished using a reactive navigation method. Chapter 5 proposes a high level data fusion method for navigation *goal detection*. A goal-driven situation assessment framework verifies the navigation goals by using sensory data and prior knowledge. Fuzzy cognitive map is used as a high level reasoning engine. Concluding remarks are given in Chapter 6.

Chapter 2

Mobile Robot Motion and Perception

2.1 Robot Motion

Robot motion has been studied thoroughly in the last few decades. Robots that have the capability of moving in their environment without being fixed to a single physical location are referred to as mobile robots. This thesis work is based on *wheeled mobile robots* (WMRs) utilized in indoor environments. WMRs are utilized in both indoor and outdoor environments. However, they are increasingly popular in industrial and research applications, particularly when flexible motion capabilities are required on reasonably smooth grounds and surfaces. Locomotion and wheel arrangement of WMR are decided based on the application it is utilized. It is not possible to use the same type of wheel arrangement with mobile robots operating in uneven outdoor terrains and planar indoor terrains. The following section discusses the motion mechanisms of WMRs operating in planar environments.

2.1.1 Wheeled Locomotion

Mobile robots employ locomotion mechanisms to move from one place to another. Depending on the application, mobile robots use different locomotion mechanisms to navigate in the environment: legged mobile robots walk, bird-like robots fly, frog-like robot jump, and fish-like robots swim. WMRs uses a high energy efficient rolling mechanism which is quite simple to control as well. This has been the most popular locomotion mechanism for mobile robots. In comparison to the locomotion mechanisms of other ground robots, wheeled robots have high stability and balance. The stability of the

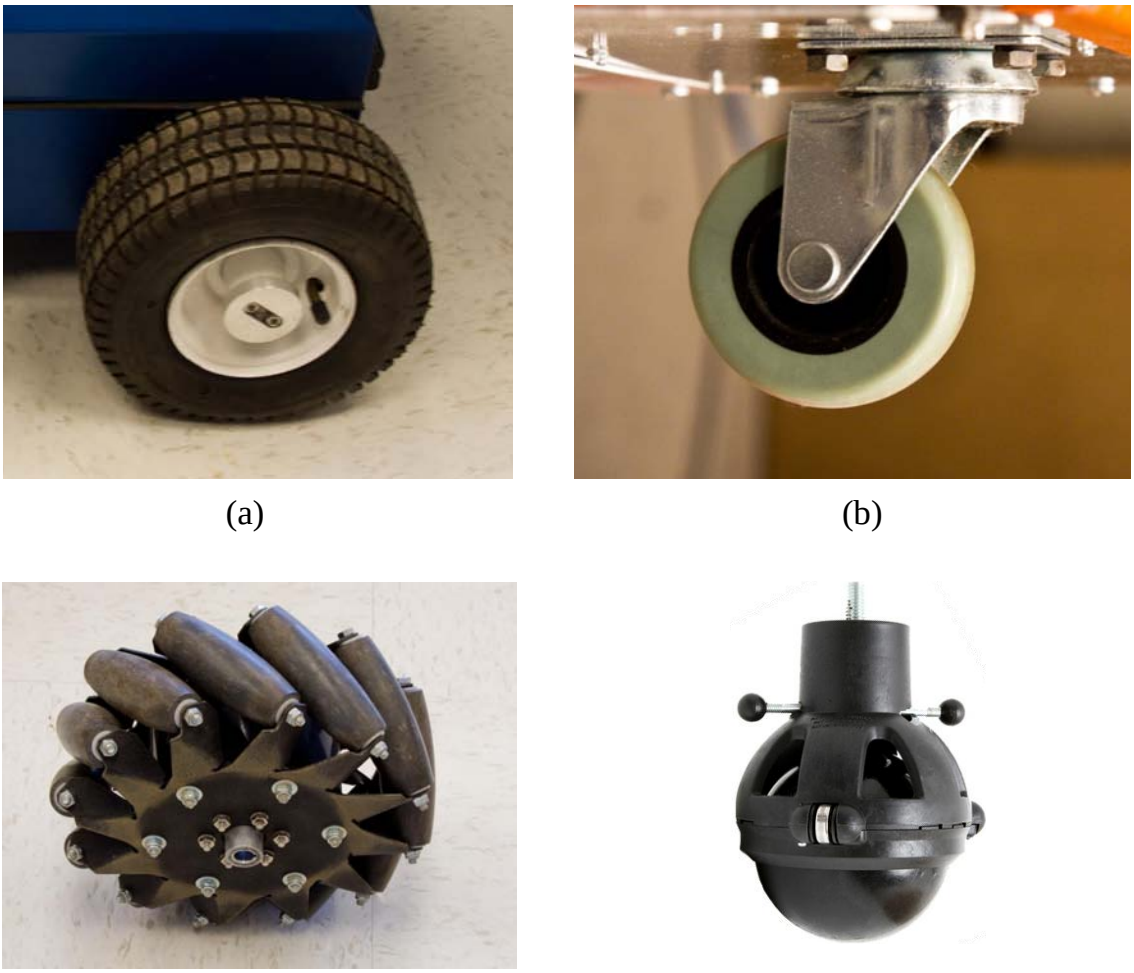


Figure 2.1: The four basic wheel types: (a) Standard wheel, (b) Castor wheel, (c) Swedish wheel, and (d) Spherical wheel.

robot can be achieved by using minimum of three wheels. Wheel types and configurations of the mobile robots are decided according to the maneuverability and controllability requirements of the robot. Maneuverability and controllability always has an inverse relationship, where highly maneuverable WMRs are less controllable and vice versa.

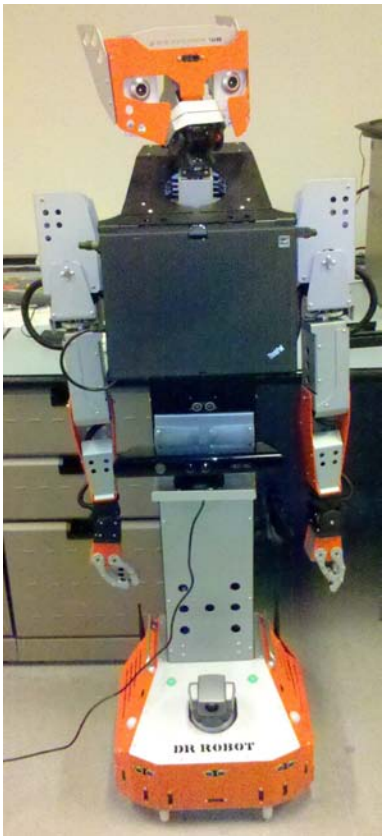
For an ideal rolling wheel, it is assumed that the wheel moves due to pure rolling and no slip occurs in any direction. According to the kinematic configurations, mainly there are four wheels types: *standard wheel*, *castor wheel*, *spherical wheel*, and *Swedish wheel* (Figure 2.1). Standard wheel can be fixed or center orientated. Any of those two

types has a primary axis of rotation. Centered oriented wheel has an added advantage of rotating along the vertical axis that goes through the center of the wheel, whereas, the standard wheel can only be rolled in one direction. Unlike the standard wheel, castor wheel rotates around an offset axis. Generally in WMRs, castors are mounted on a pivot so that it can align itself in the direction of travel. The Swedish wheel (also known as Mecanum wheel) has an extra degree of freedom compared to the standard wheel or the castor wheel, *i.e.* it can move in more than one direction. It has small rollers attached to the wheel circumference to reduce the resistance which facilitates it to move in non-conventional directions. Swedish 90 and Swedish 45 are the most popular implementations of the Swedish wheel. In Swedish 90, the small passive rollers are attached to the wheel so that their axes are orthogonal to the wheel axis which allows WMRs to easily move perpendicular to the conventional directions. In Swedish 45, those rollers are angled by 45^0 so that it has low resistance in the direction which is 45^0 angled to the conventional moving direction. The spherical or ball wheel is a omnidirectional wheel, *i.e.* it allows WMRs to move in any direction in the workspace. Actively powered rollers attached on top of wheel make it roll.

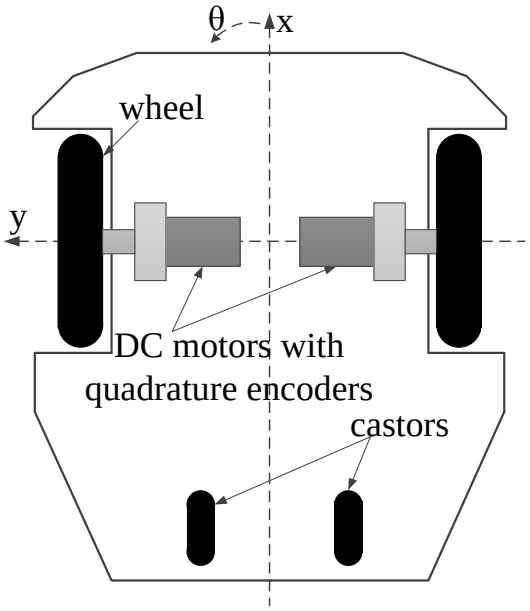
2.1.2 Wheel Arrangement

Even though the stability can be achieved with three wheels, most of the robots use four wheels or more. If more than three wheels are used for WMRs utilized in rough terrain, it needs to come up with a separate mechanism such as suspension to keep all the wheels in contact with the terrain. The number of wheels and wheel arrangement used also depend on the type of the wheels. The fundamental characteristics of WMRs, maneuverability, controllability, and stability are governed by these choices.

For an ordinary WMR, static stability can be achieved using minimum of three wheels when the center of mass is inside the triangle formed by the ground contact



(a) H20 WMR with a Kinect sensor



(b) Motion mechanism of H20 WMR.

Figure 2.2: The H20 mobile wheeled robot platform.

points of the wheels. However, stability conditions can be satisfied even with two wheels if the center of mass is below the axis connecting these two wheels. Such robots are highly maneuverable as they can move in a direction perpendicular to the conventional direction just by rotating itself about the center of the axis. WMRs with three or more wheels need to employ spherical or Swedish wheels to have this level of maneuverability. As WMRs becomes more maneuverable, they become less controllable. Omnidirectional WMRs generally comes with the wheels which has high degree of freedom (e.g. spherical wheels). This makes it more difficult to control in a specific direction. It is always advantageous to have WMRs which are highly stable, maneuverable, and controllable. However, there are no practical robot designs which can maximize all three factors simultaneously.

2.1.3 Design of the Motion System of H20 WMR

H20 (Figure 2.2(a)) is a WMR designed by DrRobot Inc. H20 is built on DrRobot's i90 robot base featuring 12" touch screen tablet featuring a 2.13GHz Intel Core i7 CPU, two large arms and dual-camera animated head. As shown in Figure 2.2(b), the H20 WMR has two standard drive wheels and two castor wheels. The two drive wheels are connected to DC motors with quadrature encoders. These encoders are capable of monitoring the revolutions and steering angles of the drive wheels. The main advantage offered by these encoders is their high resolution. The odometry is implemented using the results from these encoders. The two output channels of the quadrature encoder indicate both position and direction of rotation. Motion control of the robot is achieved by changing the wheel velocities. Two basic motion models can be used to estimate and predict the motion of the WMRs: odometry motion model and velocity motion model. The basic purpose of these motion models are to control the robot motion by calculating the change in its pose, *i.e.* $(\delta x_k, \delta y_k, \delta\theta_k)$.

2.1.4 Odometry Motion Model

Odometry motion models are basically used for estimating the robot position. Odometry is based on the readings of the wheel encoders, which are available only after executing the motion commands. Hence, it cannot be used for the motion planning of the robot. In practice, encoder readings are taken in discrete time steps. Figure 2.3 illustrates the motion of an ideal WMR in k^{th} time step. The robot pose in world coordinate frame in k^{th} time step can be denoted by (x_k, y_k, θ_k), which describes the position of the mid-axis point (x_k, y_k) and heading angle θ_k of the WMR. When the wheel base (W) is given, the change in orientation in one time step $\delta\theta_k$ can be calculated as follows

$$\delta\theta_k = \theta_k - \theta_{k-1} = \frac{(\delta r_k - \delta l_k)}{W}. \tag{2.1}$$

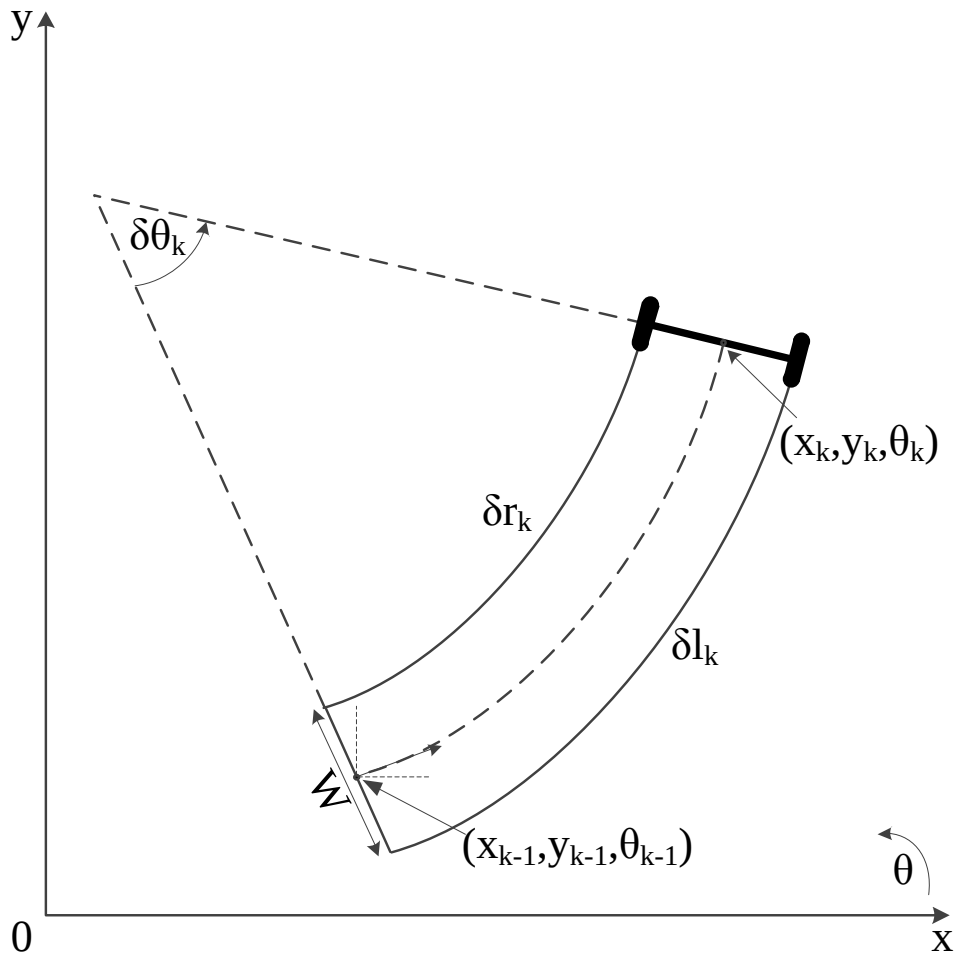


Figure 2.3: The robot motion in the world coordinate system.

Let δd_k be the distance traveled by the mid-axis point which can be calculated as the average of the distance traveled by left and right wheels,

$$\delta d_k = \frac{(\delta l_k + \delta r_k)}{2}. \tag{2.2}$$

According to Figure 2.3 and geometrical relationships, the displacement of the WMR over one time step can be calculated as follows

$$\delta x_k = x_k - x_{k-1}, \tag{2.3}$$

$$= \frac{(\delta r_k + \delta l_k)}{2} \frac{\sin(\frac{\delta\theta_k}{2})}{(\frac{\delta\theta_k}{2})} \cos(\theta_{k-1} + \frac{\delta\theta_k}{2}). \tag{2.4}$$

$$\delta y_k = y_k - y_{k-1}, \tag{2.5}$$

$$= \frac{(\delta r_k + \delta l_k)}{2} \frac{\sin(\frac{\delta\theta_k}{2})}{(\frac{\delta\theta_k}{2})} \sin(\theta_{k-1} + \frac{\delta\theta_k}{2}). \tag{2.6}$$

For a small $\delta\theta_k$;

$$\delta x_k = \delta d_k \cos(\theta_{k-1} + \frac{\delta\theta_k}{2}), \tag{2.7}$$

$$\delta y_k = \delta d_k \sin(\theta_{k-1} + \frac{\delta\theta_k}{2}). \tag{2.8}$$

Equations (2.7) and (2.8) are valid for calculating the WMR displacement during any type of robot motion for a small time period (so that $\delta\theta$ is small). There are two special cases of robot movements that we can consider under this model: straight motion and rotation about the center of the wheel axle. When the robot is moving straight, $\delta\theta \to 0$ and $\delta r_k = \delta l_k$, which simplify Equations (2.7) and (2.8) as,

$$\delta x_k = \delta r_k \cos(\theta_{k-1}), \tag{2.9}$$

$$\delta y_k = \delta r_k \sin(\theta_{k-1}). \tag{2.10}$$

When the robot is rotating around the center of the wheel axle, $(W/2)\delta\theta = \delta r_k = -\delta l_k$, results in,

$$\delta\theta = 2\frac{\delta r_k}{W} = -2\frac{\delta l_k}{W}, \tag{2.11}$$

$$\delta x = \delta y = 0. \tag{2.12}$$

According to these equations, it is possible to calculate the change of pose after the robot moved. However in reality, odometry is erroneous due to wheel slippage, misalignment and drift. Odometry error modeling is discussed by Kleeman *et al.* in [53] and [54]. A probabilistic approach of representing odometry error was introduced by Thrun *et al.* in [17].

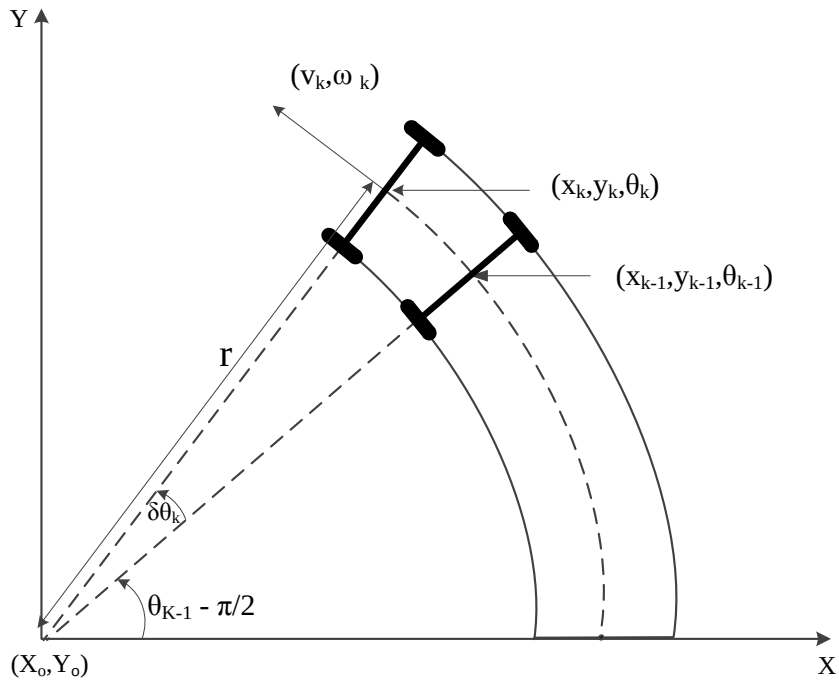


Figure 2.4: The robot motion with constant velocities ω_k and v_k.

2.1.5 Velocity Motion Model

Velocity motion models are used for the motion planning of the WMRs. Accuracy of executing velocity commands are less accurate compared to measuring the wheel revolutions. Hence the velocity motion models are generally less accurate compared to the odometry motion models. This model assumes that the motion of the robot can be controlled through two velocities: the rotational velocity and the transactional velocity. In k^{th} time step, the rotational velocity and transactional velocity can be denoted by ω_k and v_k. It is assumed that the positive transactional velocities indicates forward motion and positive rotational velocity corresponds to counterclockwise rotations.

Let $u_k = (v_k, \omega_k)$ be the control input in k^{th} time step for an ideal WMR whose motion can be perfectly controlled using the rotational and transactional velocity. Let δt_k be the duration of each time step. The change in orientation in k^{th} time step can be calculated as follows

$$\delta\theta_k = \omega_k \delta t_k. \tag{2.13}$$

If we assume that v_k and ω_k are fixed during δt_k, as illustrated in Figure 2.4, the WMR moves in a circle with radius r,

$$r = |\frac{v_k}{\omega_k}|. \tag{2.14}$$

There are two special cases that can considered under Equation (2.14): $v_k = 0$ and $\omega_k = 0$. When $v_k = 0$, the robot starts to rotate around the center of its wheel axle ($r = 0$). When $\omega_k = 0$, $r \to \infty$, *i.e.* the robot moves in a straight line.

Let, (x_o, y_o) be the center of the circle where the WMR moves in during k^{th} time step. According to the Figure 2.4,

$$x_o = x_{k-1} - \frac{v_k}{\omega_k}\sin(\theta_{k-1}), \tag{2.15}$$

$$y_o = y_{k-1} + \frac{v_k}{\omega_k}\cos(\theta_{k-1}). \tag{2.16}$$

After time δt_k, the WMR will be at (x_k, y_k, θ_k). Using simple trigonometry, current coordinates of the robot can be calculated as follows,

$$x_k = x_o + \frac{v_k}{\omega_k}\sin(\theta_{k-1} + \delta\theta_k), \tag{2.17}$$

$$y_k = y_o - \frac{v_k}{\omega_k}\cos(\theta_{k-1} + \delta\theta_k). \tag{2.18}$$

By using Equations (2.15) - (2.18), it is possible to calculate the change in robot position in k^{th} time step,

$$\delta x_k = x_k - x_{k-1}, \tag{2.19}$$

$$= -\frac{v_k}{\omega_k}\sin(\theta_{k-1}) + \frac{v_k}{\omega_k}\sin(\theta_{k-1} + \delta\theta_k). \tag{2.20}$$

$$\delta y_k = y_k - y_{k-1}, \tag{2.21}$$

$$= \frac{v_k}{\omega_k}\cos(\theta_{k-1}) - \frac{v_k}{\omega_k}\cos(\theta_{k-1} + \delta\theta_k). \tag{2.22}$$

Although the velocity motion models are very useful in motion planning and control of WMRs, they are highly erroneous in practice due to mismatch with actual motion

controllers of the robot. They also suffer from the wheel misalignments and wheel drifts similar to the odometry motion models. Thrun *et al.* have presented a probabilistic method of representing velocity motion model errors in [17].

2.2 Robot Perception

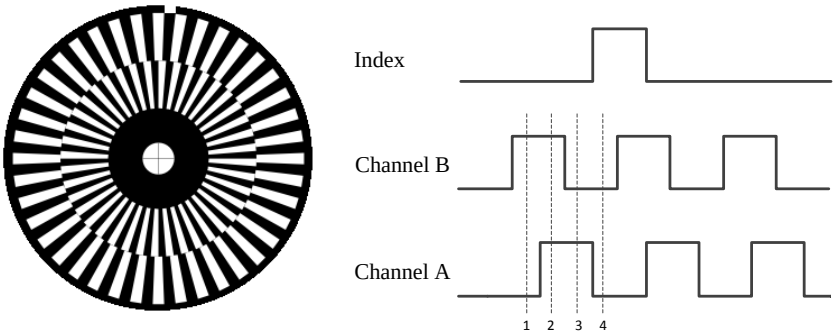
Currently mobile robots are used in many different type of environments which can vary from a familiar home environment to a surface of an extraterrestrial plant. Regardless of the type of the environment they utilize, robots need to acquire information about its workspace in order to reach their goals. This is achieved by using sensor. A sensor measures a physical quantity and converts that into a signal which can be read by the robot control program. The extraction of the required information about the environment from these readings is done after this step.

The type of sensors used in robots varies according to the type of application and its environment. The class of sensors used depends on the applications. As an example, proximity sensors are used in manipulator arms to keep desired distance between the arm and surface [55] while the mobile robot platform may use infrared (IR) sensors to follow the lines in the factory floor. However, sometimes same sensor may be used in different type of robots. As an example, cameras are used in mobile robots for collision avoidance. At the same time, they are used in close to the end effector of manipulator arms to observe the objects it is handling. As we work with a WMR, we are more interested in sensors for mobile robots. A comprehensive study about such sensors can be found in H.R. Everett's *Sensors for Mobile Robots* book [56] and Jacob Fraden's *Handbook of Modern Sensors* book.

2.2.1 Sensor Classification

Sensors used in robots are mainly classified into two categories according how they measure: *active sensors* and *passive sensors* [57]. Active sensors provide their own energy source for illumination. They emit energy which is directed towards the target to be investigated and measure the environmental reaction on that energy. Ultrasonic sensors, radar systems, and laser range finders are some examples for active sensors. These have the ability to obtain measurements anytime, regardless of the environmental condition. They are used for examining wavelengths that are not sufficiently provided by the sun, such as microwaves, or to better control the way a target is illuminated. However, active systems require the generation of a fairly large amount of energy to adequately illuminate targets. It may also interfere with other signals in the environment, specially the signals emitted from sensors in other robots or from the similar type of sensors in the same robot. The signals emitted by ultrasonic sensors are a good example in this case. Passive sensors, on the other hand, are used to detect energy in the presence of naturally occurring energy. Cameras, microphones, light and temperature sensors are some examples for passive sensors used in robots. These type of sensors cannot have a proper perception about the environment when there is no enough energy available in the environment. For example, cameras can obtain information only in the presence of adequate light.

Sensors are also classified as *proprioceptive sensors* and *exteroceptive sensors* based on what they measure [57, 58]. Proprioceptive sensors are typically passive and measure the internal state values of the robot, such as battery voltage, temperature, motor current, wheel speed, and position. Exteroceptive sensors, on the other hand, obtain the information about robot's environment such as temperature, distance to an object, sound amplitude, and light intensity. As shown in Figure 2.2, our work is mainly based on the information acquired from two sensors: *optical encoder* and *Kinect sensor*. Func-



State	Channel A	Channel B
1	high	low
2	high	low
3	low	high
4	low	high

Figure 2.5: Functionality of quadrature wheel encoder.

tionality of those two sensors are described in the next two sections.

2.2.2 Optical Encoder

The optical encoders were first developed in the mid-1940s by Baldwin Piano Company [59]. They were used as *tone wheels* that allowed electric organs to mimic other musical instruments. Currently in mobile robotics field, it has become the most popular mechanism of measuring the angular position and speed of the motor shaft. The optical encoder is a proprioceptive sensor as it is used to measure an internal state of the robot. They can correctly estimate the position in the frame of robot. Mobile robots have heavily benefited from the high-resolution, low-cost wheel encoders available in the current market.

Any optical encoder basically contains of a light source, a matched photo detector, and a rotor disc with a fine optical grid that rotates with the motor shaft. A focused beam of light aimed at the photo detector is periodically interrupted by the coded optical grid attached to the motor shaft. This results in a certain number of sine waves for each shaft revolution. The resulting sine wave is converted into a discrete square wave using a threshold value defined between the light and dark states.

Single-channel optical encoders are incapable of determining the direction. Hence they cannot be used as position sensors of the WMRs. Quadrature encoders overcome

Figure 2.6: Kinect sensor.

this issue by adding a second channel. In these encoders, a second illumination and detector pair is placed 90° shifted with respect to the original pair so that the resulting pulse trains are 90° out of phase as shown in Figure 2.5. It is possible to determine the direction of rotation of the wheel using this technique by identifying which square wave produces the rising edge first. Index output in the outer channel produces a reference pulse for each revolution.

Encoder resolution is measured in *counts per revolution* (CPR) [60]. CPR rating can be improved by a factor of four using four different states in quadrature encoders. DrRobot's H20's quadrature encoders yield 400 CPR and HAWK's quadrature encoders yield 800 CPR. The minimum angular resolution of the optical encoder can be calculated from its CPR rating. Also the actual moving distance of the drive wheels can be computed from the encoder readings and its CPR rating.

$$distance = \frac{2\pi R}{(CPR)}\Delta e. \tag{2.23}$$

Here Δe is the associated encoder counts and R is the radius of the drive wheel.

2.2.3 Kinect Sensor

Kinect (Figure 2.6) is a motion sensing input device introduced by Microsoft for video gaming. It enables users to control the game with physical gestures and voice-based

commands. Lately it has been a very popular sensor among the robotics research community [61, 62]. In mobile robotics, it plays a significant role as an exteroceptive sensor that replaces the ultrasonic sensors and laser range finders. Both ultrasonic sensor and laser range finder are based on the time-of-flight, a technique which uses the propagation speed of the emitted signal to measure the distance to the objects in its environment. This touch-free sensor is powered by an RGB camera and a depth sensor. Figure 2.7 shows the RGB and and depth images captured by a Kinect sensor.

Unlike traditional range finders, the Kinect depth image is acquired using the Light Coding technology [63]. The coded light is captured by the IR camera in order to produce the Kinect disparity matrix. The relationship between the Kinect disparity and actual depth value is given by,

$$z = \frac{b \times f}{\frac{1}{8}(\mu - d_{kinect})}, \tag{2.24}$$

where z is the actual depth, b is the distance between the IR camera and laser-based IR projector lenses which is about 7.5 cm, f is the focal length of the IR camera in pixels which is typically 580, d_{kinect} is the Kinect disparity which provides 2048 levels of sensitivity in VGA resolution with 11-bit depth, and μ is an offset value for a given Kinect device. The factor 1/8 is used due to the fact that d_{kinect} is in 1/8 pixel units. Unlike ordinary stereo pairs, the actual depth does not become infinity at the zero Kinect disparity.

In Equation 2.24, the value for μ needs to be estimated using a calibration process. Our calibration setup is shown in Figure 2.8. The experimental values are obtained by manually measuring the physical distance to the target object (green circle) from the Kinect sensor. The target object is moved away from the Kinect sensor by 10cm each time and corresponding Kinect disparity values are obtained. Figure 2.9 illustrates the Kinect depth calibration results. The theoretical curve represents the results obtained using Equation (2.24). The value for μ is adjusted so that the experimental values

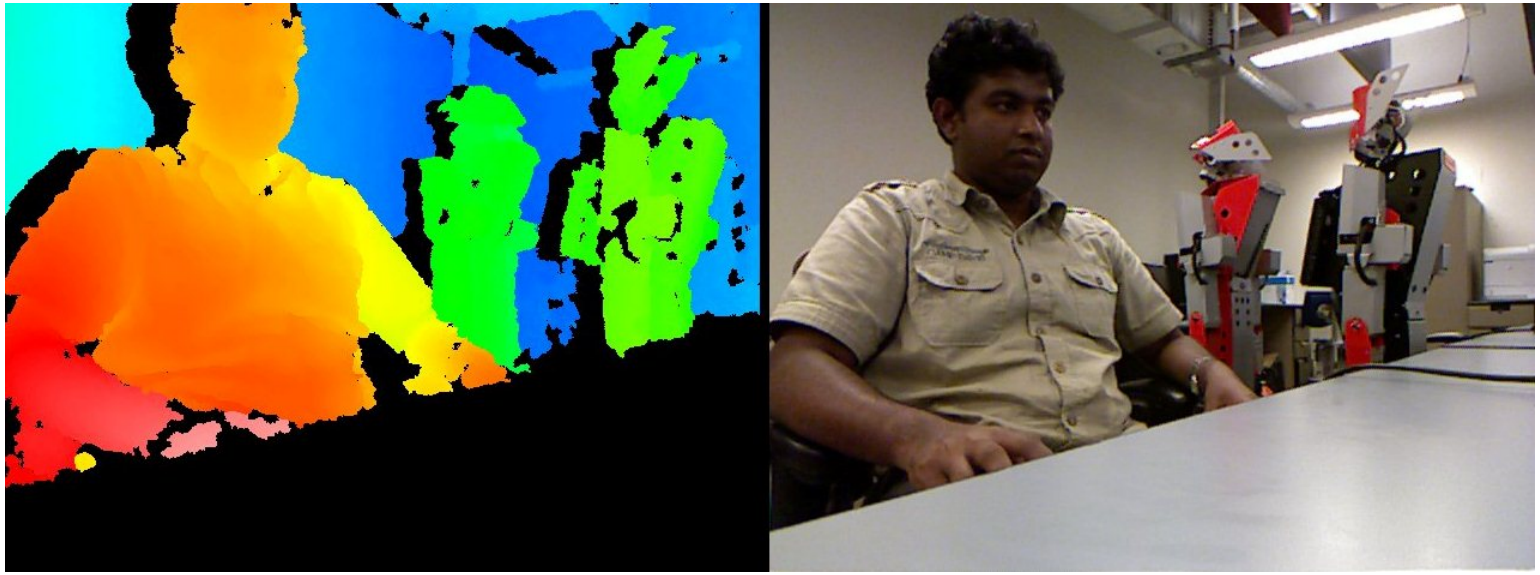


Figure 2.7: RGB image (right) and corresponding depth map (left) captured by a Kinect device.

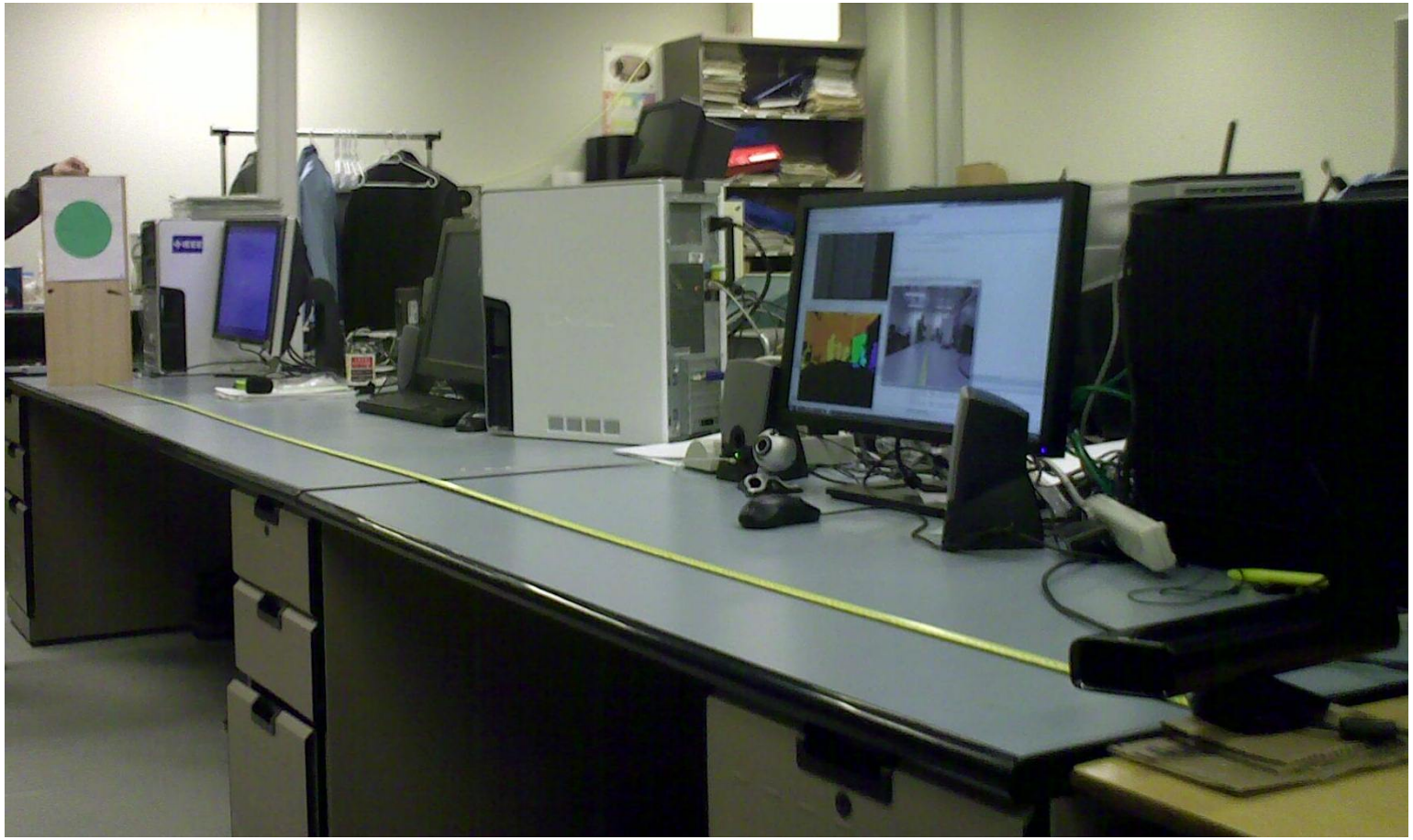


Figure 2.8: Kinect calibration setup.

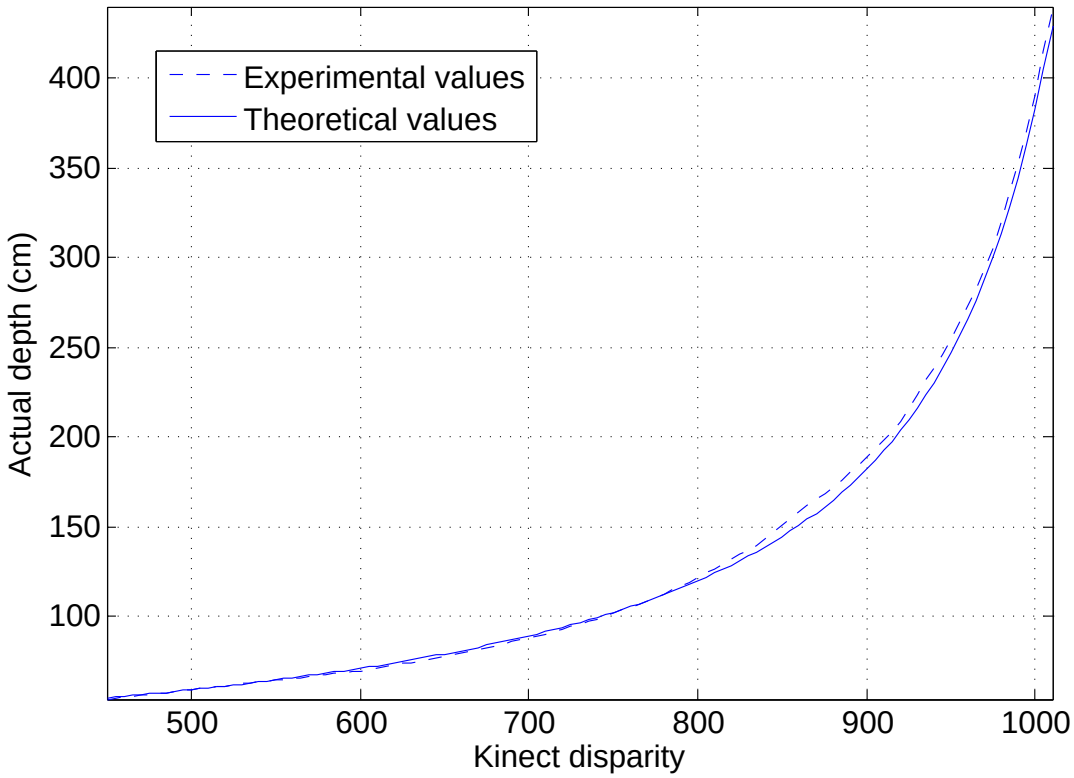


Figure 2.9: Kinect depth calibration results.

coincide with the theoretical values. According to our calibrations, the value for μ is 1091.50. However this calibration results may slightly differ from one sensor to another. Therefore each sensor should be calibrated separately before using it for depth estimation. Although the Kinect sensor has obtained huge popularity in recent past, it has its own limitations.

The Kinect sensors has an operation range of 0.8 m – 3.5 m with the resolution of 1 cm at a distance of 2 m [63]. The Hokuyo URG-04LX-UG01 scanning laser range finder works from 0.06 m to 4 m with 1% error [64], and more advanced Hokuyo UTM-30LX scanning laser range finder works from 0.1 m to 60 m [65], *i.e.* it can sense objects nearly 56m before Kinect can detect it. This is a huge advantage in large and dynamic environments. However these more expensive sensors also face the problem of close range blind spot as observed previously with the Kinect sensor. This can cause many problems

in dynamic environments, especially with the Kinect's narrow *field-of-view* (FOV). The depth image on the Kinect has a field of view of 58° [63], whereas the Hokuyo URG-04LX-UG01 laser comes with 240° FOV [64] and UTM-30LX with 270° [65]. It will be hard to detect if a dynamic obstacle approach from behind the Kinect and stay within its blind region (< 0.8 m). FOV can be increased by using more than single Kinect device. However, the main problem with this approach is the sheer volume of data. 640 x 480 x 30 fps x (3 bytes of color + 2 bytes of Depth) puts us at close to the maximum speed of the universal serial bus 2.0 (USB 2.0), at least to the point where you are only going to get good performance with one Kinect per bus. FOV can be also increased by attaching a servo motor to pan the Kinect horizontally, or else, rotating the robot with the Kinect on its own axis of rotation.

Compared to its closest counterpart, the laser range finder, the Kinect is considerably a cheap sensor. It also has a high horizontal angular resolution of 0.1^o, whereas the Hokuyo URG-04LX-UG01 laser comes with 0.36^o FOV [64] and UTM-30LX with 0.25^o [65]. On the other hand, ultrasonic sensors are even cheaper than the Kinect sensor. Parallax Ping ultrasonic ranger has a detection range of 2 cm - 3 m[66]. However, ultrasonic range finders are becoming less popular among the robot community due to their low resolution, high cross sensitivity, and low bandwidth (50 Hz).

2.3 Summary

To summarize, in this chapter, the key concepts of mobile robot motion and perception which are required for the complete understanding of the novel work presented in this thesis have been reviewed. The basic concepts of robot motion relevant to this work have been discussed in Section 2.1. The locomotion and different wheel arrangement of the WMRs have been introduced in Subsections 2.1.1 and 2.1.2. The odometry and velocity motion models used in this thesis have been introduced in Subsections 2.1.4

and 2.1.5. An overview of the classification of sensors used in mobile robotics has been given in Subsection 2.2.1 followed by more detailed discussions on the sensors utilized in this work: Optical encoders and the Kinect Sensor. The functionality of the optical encoders has been presented in Subsection 2.2.2. The calibration of the Kinect sensor and a comparison of the Kinect sensor with its counterparts are introduced in Subsection 2.2.3.

Chapter 3

Indoor Mobile Robot Localization

3.1 Introduction

Mobile robot localization is one of the fundamental problems in mobile robot navigation and motion planning. It is an instance of general localization problem. It involves one simple question: where is the robot now? Although a simple question, answering it is not easy due to the nature of the environment and robot itself. In an indoor environment with a flat floor plan, localization is identified as a problem of estimating the pose, *i.e.* position and orientation of a mobile robot, when the map of the environment, sensor readings, and executed actions of the robot are provided [67]. Dead reckoning is the process of calculating current pose using previously determined pose and some internal measures of velocity, acceleration, and time [68, 69]. In most of the *wheeled mobile robots* (WMRs), this is achieved using odometry. The encoders mounted on the wheels provide robot motion information to update the mobile robot pose. However, odometric pose estimation unboundedly accumulates errors due to different wheel diameters, wheel-slippage, wheel misalignment, and finite encoder resolution [53]. Experiment results presented in this chapter, together with previous studies [70], concur that the largest factor in odometry error is due to the rotation of the robot.

This chapter proposes an accurate and low cost mobile robot localization method using odometry with a Kinect sensor. The odometry and Kinect sensor measurements are fused using an *extended Kalman filter* (EKF) and a *Particle filter* (PF) to provide more accurate localization results. The correct detection of landmarks by applying Hough transform and depth estimation using the Kinect sensor have significantly con-

tributed towards a better performance of the robot localization. The experiments are carried out with H20 mobile robot (*see* Figure 2.2(a)) and results are provided in order to interpret the accuracy of the proposed method. A shorter version of this text is presented in the IEEE International Conference on Emerging Signal Processing Applications (ESPA'2012) [61].

3.2 Localization using Extended Kalman Filter

Kalman filtering is a commonly used approach for reducing the error in measurements from different sources [71, 72]. In EKF, unlike in basic Kalman filtering, the state transition and observation models can be non-linear functions of the state. Therefore, the EKF is widely adopted in the robot community. It is applicable to non-linear systems where the associated uncertainties are assumed to be Gaussian [17]. In this work, the mobile robot localization is achieved by fusing odometry information with Kinect measurements of the landmarks using EKF.

The system state $\mathbf{x}_k$ and observation of the state $\mathbf{z}_k$ at time step k are modeled by a non-linear system function $\mathbf{f}$ and measurement function $\mathbf{h}$ as follows

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{q}_{k-1}, \tag{3.1}$$

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_{k-1}) + \mathbf{r}_k, \tag{3.2}$$

where the state vector $\mathbf{x}_k = [x_k \; y_k \; \theta_k]^T$. The input vector $\mathbf{u}_k = [\delta l_k \; \delta r_k]^T$, which indicates the distances traveled by the left and right wheels, respectively, are used as inputs in the kinematic model. Parameters $\mathbf{q}_{k-1}$ and $\mathbf{r}_k$ are the system and measurement noises.

3.2.1 Prediction Step

In the prediction step, the EKF predicts the future state of the system $\hat{\mathbf{x}}_k^-$ and the state error covariance matrix $\mathbf{P}_k^-$ such that

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \hat{\mathbf{u}}_k), \tag{3.3}$$

$$\mathbf{P}_k^- = \nabla\mathbf{f}_{\mathbf{x}_{k-1}}\mathbf{P}_{k-1}\nabla\mathbf{f}_{\mathbf{x}_{k-1}}^T + \nabla\mathbf{f}_{\mathbf{u}_k}\mathbf{U}_k\nabla\mathbf{f}_{\mathbf{u}_k}^T + \mathbf{Q}_{k-1}, \tag{3.4}$$

where $\mathbf{U}_k$ and $\mathbf{Q}_{k-1}$ are the covariances of the input and system noises, respectively. The Jacobians of $\mathbf{f}$ with respect to $\mathbf{x}_{k-1}$ and $\mathbf{u}_k$ at the point $(\hat{\mathbf{x}}_{k-1}, \hat{\mathbf{u}}_k)$ are defined as

$$\nabla\mathbf{f}_{\mathbf{x}_{k-1}} = \frac{\partial\mathbf{f}}{\partial\mathbf{x}_{k-1}}|_{(\hat{\mathbf{x}}_{k-1},\hat{\mathbf{u}}_k)} \text{ and } \nabla\mathbf{f}_{\mathbf{u}_k} = \frac{\partial\mathbf{f}}{\partial\mathbf{u}_k}|_{(\hat{\mathbf{x}}_{k-1},\hat{\mathbf{u}}_k)}. \tag{3.5}$$

From the odometry model described in Section 2.1.4, the state of the system can mentioned in Equation (3.3) can be predicted as

$$\mathbf{f}(\hat{\mathbf{x}}_{k-1}, \hat{\mathbf{u}}_k) = \begin{bmatrix} \hat{x}_{k-1} + \delta d_k \cos(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ \hat{y}_{k-1} + \delta d_k \sin(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ \hat{\theta}_{k-1} + \delta\theta_k \end{bmatrix}. \tag{3.6}$$

Using Equations (3.5) and (3.6), $\nabla\mathbf{f}_{\mathbf{x}_{k-1}}$ and $\nabla\mathbf{f}_{\mathbf{u}_k}$ can be calculated:

$$\nabla\mathbf{f}_{\mathbf{x}_{k-1}} = \begin{bmatrix} 1 & 0 & -\delta d_k \sin(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ 0 & 1 & \delta d_k \cos(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.7}$$

$$\nabla\mathbf{f}_{\mathbf{u}_k} = \begin{bmatrix} \hat{x}_{k-1} + \delta d_k \cos(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ \hat{y}_{k-1} + \delta d_k \sin(\hat{\theta}_{k-1} + \frac{\delta\theta_k}{2}) \\ \hat{\theta}_{k-1} + \delta\theta_k \end{bmatrix}. \tag{3.8}$$

Figure 3.1: Distribution of the artificial landmarks in the environment.

3.2.2 Landmark Detection with Kinect Sensor

In the update step, the WMR uses landmarks around it to estimate its pose. In this work, different colored circles are used as landmarks as shown in Figure 3.1. The images and their depth values are acquired by the Kinect sensor. The OpenKinect library was used in order to obtain data from Kinect sensor [73]. Hough transform filters are used to detect the landmarks (circles) in RGB image frames [74]. The landmarks are distinguished from each other using HSI color model.

The observation of the state $\mathbf{z}_k$ can be expressed using the measurements obtained from the Kinect sensor,

$$\mathbf{z}_k = \begin{bmatrix} \alpha_k & \lambda_k \end{bmatrix}^T. \tag{3.9}$$

As illustrated in Figure 3.2, the azimuth angle α_k with respect to the WMR x-axis and the distance λ_k to the ith landmark $B_i(x_{B_i}, y_{B_i})$ at a time instant k can be used to determine the value of the measurement function,

$$\mathbf{h}(\hat{\mathbf{x}}_k^-) = \begin{bmatrix} \tan^{-1}(\frac{y_{B_i}-\hat{y}_k^-}{x_{B_i}-\hat{x}_k^-}) - \hat{\theta}_k^- \\ \sqrt{(x_{B_i}-\hat{x}_k^-)^2 + (y_{B_i}-\hat{y}_k^-)^2} \end{bmatrix}. \tag{3.10}$$

3.2.3 Update Step

Once the measurement $\mathbf{z}_k$ is available, the optimal Kalman gain matrix $\mathbf{K}_k$ can be determined as follows

$$\boldsymbol{\nu}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-), \tag{3.11}$$

$$\mathbf{S}_k = \nabla\mathbf{h}_{\mathbf{x}_k}\mathbf{P}_k^-\nabla\mathbf{h}_{\mathbf{x}_k}^T + \mathbf{R}_k, \tag{3.12}$$

$$\mathbf{K}_k = \mathbf{P}_k^-\nabla\mathbf{h}_{\mathbf{x}_k}^T\mathbf{S}_k^{-1}, \tag{3.13}$$

where $\boldsymbol{\nu}_k$ is the innovation matrix which is assumed to be corrupted by zero-mean Gaussian noise with covariance $\mathbf{S}_k$. Here, $\mathbf{S}_k$ and $\mathbf{R}_k$ represent the innovation covariance matrix and measurement covariance matrix, respectively. The Jacobian $\nabla\mathbf{h}_{\mathbf{x}_k}$ is given

by,

$$\nabla \mathbf{h}_{\mathbf{x}_k} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}_k}|_{\hat{\mathbf{x}}_k^-}, \tag{3.14}$$

$$= \begin{bmatrix} \frac{\partial h_{\alpha_k}}{\partial x_k} & \frac{\partial h_{\alpha_k}}{\partial y_k} & \frac{\partial h_{\alpha_k}}{\partial \theta_k} \\ \frac{\partial h_{\lambda_k}}{\partial x_k} & \frac{\partial h_{\lambda_k}}{\partial y_k} & \frac{\partial h_{\lambda_k}}{\partial \theta_k} \end{bmatrix}. \tag{3.15}$$

Here,

$$\frac{\partial h_{\alpha_k}}{\partial x_k} = \frac{y_{B_i} - y_k}{(x_{B_i} - x_k)^2 + (y_{B_i} - y_k)^2}, \tag{3.16}$$

$$\frac{\partial h_{\alpha_k}}{\partial y_k} = \frac{-x_{B_i} + x_k}{(x_{B_i} - x_k)^2 + (y_{B_i} - y_k)^2}, \tag{3.17}$$

$$\frac{\partial h_{\alpha_k}}{\partial \theta_k} = -1, \tag{3.18}$$

$$\frac{\partial h_{\lambda_k}}{\partial x_k} = \frac{-x_{B_i} + x_k}{\sqrt{(x_{B_i} - x_k)^2 + (y_{B_i} - y_k)^2}}, \tag{3.19}$$

$$\frac{\partial h_{\lambda_k}}{\partial y_k} = \frac{-y_{B_i} + y_k}{\sqrt{(x_{B_i} - x_k)^2 + (y_{B_i} - y_k)^2}}, \tag{3.20}$$

$$\frac{\partial h_{\lambda_k}}{\partial \theta_k} = 0. \tag{3.21}$$

The measurement update is done using the optimal Kalman gain as calculated in Equation (3.13).

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \boldsymbol{\nu}_k, \tag{3.22}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T, \tag{3.23}$$

where $\hat{\mathbf{x}}_k$ is the state estimate at the time step k.

3.2.4 EKF Realization

The initial state estimate is taken as $\mathbf{x}_k = \mathbf{0}$ and $\mathbf{P}_k = \mathbf{0}$, *i.e.* the initial vehicle pose defines the base coordinate frame. The measurement update step of the EKF takes place only when a landmark is detected. Whenever a landmark is not detected, the predicted state and state error covariance matrix of the time update step are taken as the state

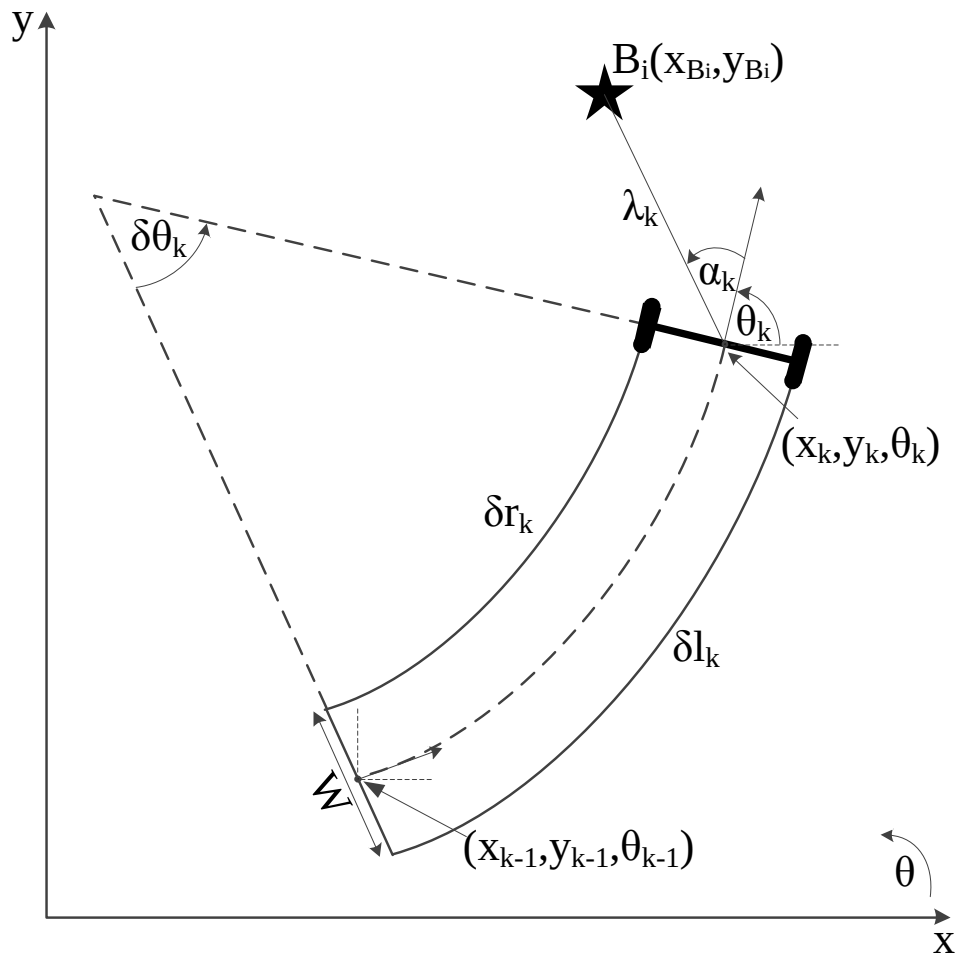


Figure 3.2: The robot coordinate system and position of ith landmark.

estimate and state error covariance matrix for the next iteration of the filter, $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^-$ and $\mathbf{P}_k = \mathbf{P}_k^-$.

Here, the system, measurement and input noises are assumed to be zero mean and uncorrelated. Hence, the noise covariance matrices, $\mathbf{Q}_k$, $\mathbf{R}_k$, and $\mathbf{U}_k$ become diagonal. The system and measurement noises are also assumed to be time invariant, which leads $\mathbf{Q}_k$ and $\mathbf{R}_k$ to be time invariant as well [21].

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_{x_k}^2 & 0 & 0 \\ 0 & \sigma_{y_k}^2 & 0 \\ 0 & 0 & \sigma_{\theta_k}^2 \end{bmatrix}, \tag{3.24}$$

$$\mathbf{R}_k = \begin{bmatrix} \sigma_{\alpha_k}^2 & 0 \\ 0 & \sigma_{\lambda_k}^2 \end{bmatrix}. \tag{3.25}$$

The system position noise variances of (x, y) coordinates are denoted by $\sigma_{x_k}^2$ and $\sigma_{y_k}^2$. Here, $\sigma_{\theta_k}^2$ is the orientation noise variance, while $\sigma_{\alpha_k}^2$ and $\sigma_{\lambda_k}^2$ are the measurement noise variances.

The variance of the noise generated by each encoder can be determined as the sum of the variance of each independent unit because the encoder measurements are statistically independent and they accumulate errors over time. Therefore it is possible to assume that the variance of each unit of travel is proportional to the total distance traveled [53].

$$\mathbf{U}_k = \begin{bmatrix} \sigma_L^2 & 0 \\ 0 & \sigma_R^2 \end{bmatrix}, \tag{3.26}$$

where,

$$\sigma_L^2 = k_L^2 \sum_{i=0}^{k} |\delta l_i|, \tag{3.27}$$

and

$$\sigma_R^2 = k_R^2 \sum_{i=0}^{k} |\delta r_i|. \tag{3.28}$$

Here, k_L^2 and k_R^2 are positive constants.

3.3 Localization using Particle Filter

As a solution to the Gaussian density assumption inherent in EKF localization, Monte Carlo localization was introduced for mobile robots [22]. It can globally localize the robot due to its capability of representing multi-modal distributions. The basis of the PF localization is to create a sample-based representation of the entire distribution of the robot trajectory. It requires less amount of memory and returns more accurate results compared to grid-based Markov localization [22]. The uncertainty in state estimation is represented by a set of samples that are randomly drawn from the probability density function, which are also known as particles.

Similar to the EKF localization, this PF localization also has predication and update steps which are evaluated recursively. If a landmark is detected, particle weights are re-evaluated based on the Kinect measurements. In addition to the prediction and update steps common to both the EKF and PF, the PF performs a resampling step. It avoids the depletion of the sample population after few iterations. In this application, the system state $\mathbf{x}_k = [x_k, y_k, \theta_k]$ or the pose of the robot is modeled by set of M particles $\{\mathbf{x}_k^{[i]}\}_{i=1}^{M}$, and associated importance weights $\{w_k^{[i]}\}_{i=1}^{M}$. $w_k^{[i]}$ defines the contribution of ith particle to the overall estimate of the variable.

3.3.1 Prediction Step

In the prediction step, the particles are modified after each action according to the odometry model described in Section 2.1.4, including the addition of random noise in order to simulate the effect of noise:

$$\mathbf{x}_k^{[i]} = \begin{bmatrix} \hat{x}_{k-1} + (\delta d_k + n_{d_k}) \cos(\hat{\theta}_{k-1} + \frac{\delta\theta_k + n_{\theta_k}}{2}) \\ \hat{y}_{k-1} + (\delta d_k + n_{d_k}) \sin(\hat{\theta}_{k-1} + \frac{\delta\theta_k + n_{\theta_k}}{2}) \\ \hat{\theta}_{k-1} + \delta\theta_k + n_{\theta_k} \end{bmatrix}, \; i = 1, 2, \ldots, M. \tag{3.29}$$

Here, n_{d_k} is translational noise which is assumed to be additive Gaussian noise with zero mean and standard deviation $\sigma_{trns}\delta d_k$, *i.e.* $n_{d_k} \sim N(0, (\sigma_{trns}\delta d_k)^2)$. n_{θ_k} is rotational noise which is assumed to be additive Gaussian noise with zero mean and standard deviation $\sigma_{rot}\delta d_k$, *i.e.* $n_{\theta_k} \sim N(0, (\sigma_{rot}\delta\theta_k)^2)$. The values of the σ_{trns} and σ_{rot} need to be calculated experimentally. According to the Equation (3.29), x_k and y_k depend on both translational and rotational errors. During the forward translation of the robot, its orientation also changes resulting in deviation from the desired direction of the translation, which is generally referred to as drifting of the WMRs.

3.3.2 Update Step

If the landmarks are detected, importance weights are re-evaluated in the update step based on the sensory information in order to accurately describe posterior over robot pose. Sensor observation $\mathbf{z}_k$ can be expressed by,

$$\mathbf{z}_k = \begin{bmatrix} \alpha_k & \lambda_k \end{bmatrix}^T, \tag{3.30}$$

which is obtained using the method explained in Section 3.2.2. As illustrated in Figure 3.2, the azimuth angle α_k with respect to the WMR x-axis and distance λ_k to the ith landmark $B_i(x_{B_i}, y_{B_i})$ at a time instant k can be used to determine the value of the measurement function:

$$\mathbf{h}_k^{[i]} = \begin{bmatrix} h_{k,1}^{[i]} \\ h_{k,2}^{[i]} \end{bmatrix}, \tag{3.31}$$

$$= \begin{bmatrix} \tan^{-1}(\frac{y_{B_i} - y_k^{[i]}}{x_{B_i} - x_k^{[i]}}) - \theta_k^{[i]} \\ \sqrt{(x_{B_i} - x_k^{[i]})^2 + (y_{B_i} - y_k^{[i]})^2} \end{bmatrix}, \; i = 1, 2, \ldots, M. \tag{3.32}$$

In contrast to the measurement function explained in Section 3.2.2, here, we obtain M different values for it corresponding to each particle.

Once the Kinect sensor measurements are available, the importance weights of the particles are updated as;

$$w_k^{[i]} = \frac{1}{\sqrt{2\pi}\sigma_\alpha} e^{-\frac{(\delta\alpha_k^{[i]})^2}{2\sigma_\alpha^2}} \times \frac{1}{\sqrt{2\pi}\sigma_\lambda} e^{-\frac{(\delta\lambda_k^{[i]})^2}{2\sigma_\lambda^2}} \times w_{k-1}^{[i]},\ i = 1, 2, \ldots, M. \tag{3.33}$$

where,

$$\begin{aligned} \delta\alpha_k^{[i]} &= \alpha_k - h_{k,1}^{[i]}, \\ \delta\lambda_k^{[i]} &= \lambda_k - h_{k,2}^{[i]}. \end{aligned} \tag{3.34}$$

According to Equations (3.33) and (3.34), particles obtain higher importance weights as the corresponding values for $\delta\alpha_k^{[i]}$ and $\delta\lambda_k^{[i]}$ get smaller. Once the particle weights are calculated, they are normalized so that the total of the importance weights is equal to 1 (*i.e.* $\sum_{i=0}^{M} w^{[i]} = 1$). In the update step, the estimated pose is obtained either by using the weighted mean,

$$\hat{\mathbf{x}}_k = \sum_{i=0}^{M} w^{[i]} x_k^{[i]} \tag{3.35}$$

or by selecting the particle with highest weight as the *best particle*.

3.3.3 Resampling

After several iterations of the particle filter, the importance weights of most of the particles get close to zero, *i.e.* a small contribution to the posterior distribution of the robot pose. In literature, this effect is also known as the particle depletion problem [75]. In order to make sure that the particles represents the true posterior distribution, it is required to resample the particles at every time step. However resampling in every time step might lead to the loss of diversity in particles representing the robot path. In this work, we use *effective sample size* (ESS) [76] to measure the diversity of the particles. The ESS is calculated at the each iteration. If the calculated ESS value is

below a certain percentage of the number of particles, near-zero-weight particles are replaced by duplicating the ones with higher weights. This is commonly known as particle resampling. There are several different methods that have been proposed for resampling [77, 76, 78]. In this application, we use *linear time resampling* [78] method to eliminate the near-zero-weight particles.

3.4 Experiment Results

The experiments were carried out in an indoor environment of 7m×2m with 4 landmarks. The starting point and 5 way points were marked on the floor with exact measurements. The WMR was maneuvered through the way points and measurements were taken.

Figure 3.3 illustrates the statistical results obtain from 20 individual experiments. The ellipses around the mean values represent the standard deviations of the results. The mean values of the estimated results are close to the way points while their standard deviations are very small. It confirms that the proposed methods are capable of providing more accurate localization results compared to odometry. It should be noted that some pose errors, especially the errors in y-direction were due to the deviations in maneuvering the WMR onto the way points. Therefore the actual errors are expected to be slightly smaller. Since all methods under test are affected by such errors, the results remain qualitatively unchanged.

The root mean squared error (RMSE) of localization at each way point is shown in Figure 3.4. According to the given results, the RMSE error of odometry measurement keeps increasing considerably with the distance traveled. This leads to an erroneous pose estimation of the WMR. However the RMSE has been reduced substantially using the EKF and PF with the Kinect sensor. This proves that the proposed estimation methods are robust and stable, while the results obtained using the PF are slightly more stable compared to the EKF.

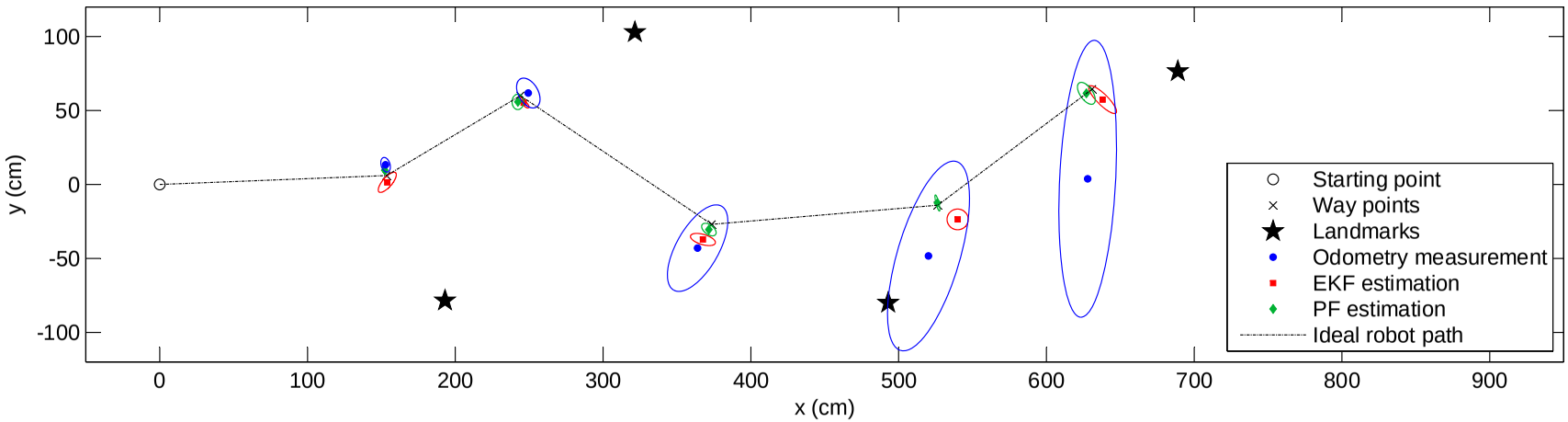


Figure 3.3: Standard deviations of the localization results.

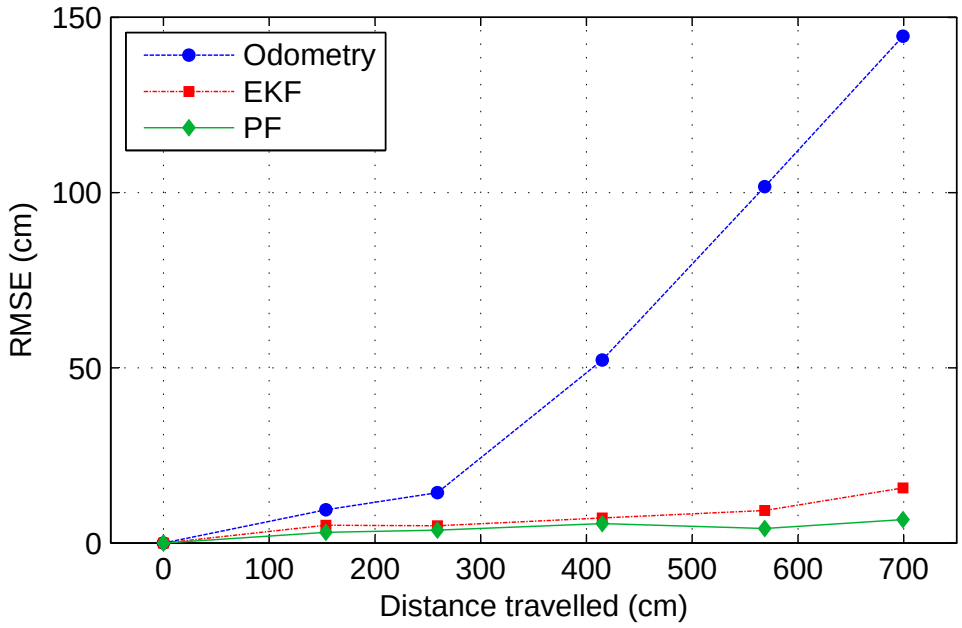


Figure 3.4: Root mean squared localization error.

Table 3.1: Euclidean distance error at each way point (cm).

Way point	1	2	3	4	5
Odometry	7.3634	5.6828	18.6643	34.9313	60.7425
EKF	4.5655	5.1575	11.7987	16.5151	10.0205
PF	3.9186	4.4614	3.9726	1.4753	4.8919

The Euclidean distance error at each way point is given in Table 3.1. These values were obtained by calculating the Euclidean distance between the way points and the average of the measured/estimated robot positions. The errors in estimation results are significantly lower compared to the errors in odometry measurements. This suggests that the proposed methods using the Kinect sensor can perform accurate mobile robot localization. According to the given results, the PF pose estimation is more accurate compared to the EKF pose estimation with the Kinect sensor.

3.5 Summary

The odometric pose estimation accumulates errors with time. More precise mobile robot localization can be achieved by reducing the uncertainty in the odometric pose estimation using a Kinect sensor to observe landmarks in an environment. Extended Kalman filter and particle filter can be used for the sensor fusion in pose estimation. The proposed measurement model for the Kinect sensor together with odometry model is capable of providing an accurate system model for a wheeled mobile robot. A robust and accurate mobile robot localization method using an inexpensive sensor system was proposed, implemented, and tested on the H20 mobile robot.

Chapter 4

Autonomous Exploration in Unknown Environments

4.1 Introduction

Mobile robot exploration is a task of controlling a robot in order to maximize its knowledge about the external environment using its sensors [17]. It has many application areas such as desert exploration [79], underwater exploration [80, 81], volcano exploration [82, 83], and outer planet exploration [10, 11]. This work mainly focuses on indoor robot exploration. Robotic exploration can be described using three subtasks, namely, *map building*, *localization*, and *motion control*. Robotic mapping is identified as a problem of creating the spatial models of physical environments using mobile robots [84]. In an indoor environment with a flat floor plan, localization estimates the pose, *i.e.* position and orientation of a mobile robot, provided that the map of the environment, sensor readings, and executed actions of the robot are given [67]. Motion control is a task of steering the robot in order to efficiently guide it to a desired location [85]. Therefore, exploration should be achieved through an integrated system which considers localization, mapping and action selection simultaneously.

Here, the exploration problem is addressed within the context of real-time navigation of non-holonomic mobile robots in unknown and uncertain environments. The architecture of the proposed exploration system is illustrated in Figure 4.1. An odometry system is used to obtain the robot motion information and Microsoft Kinect sensor [63] is used as a depth sensor to observe the environment. The issues caused by imperfect sensing is discussed in this chapter and the corresponding solutions are proposed. Information fusion of sensory data is achieved with the *Rao-Blackwellied particle filter* (RBPF) with

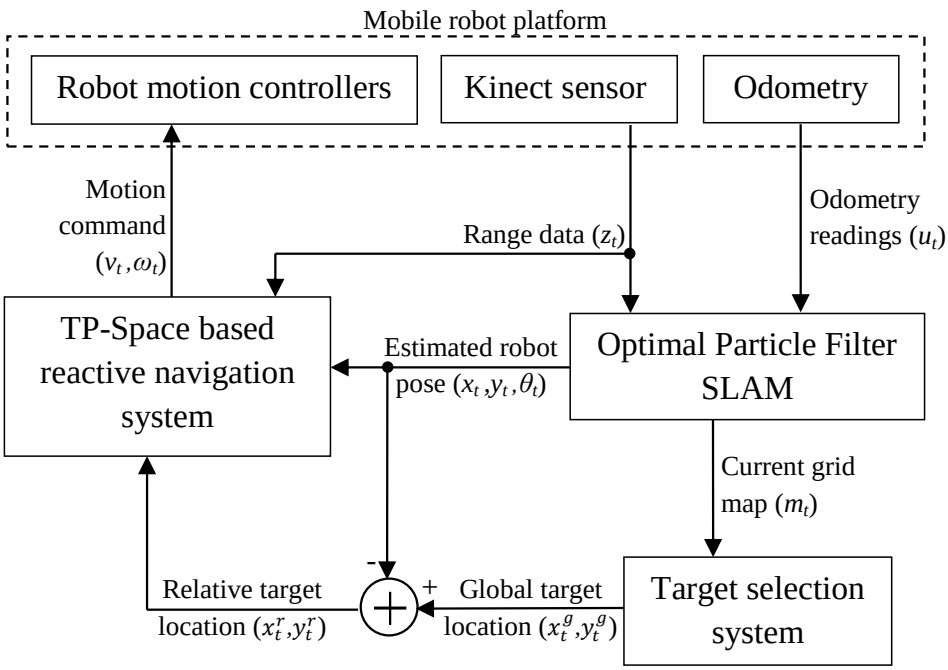


Figure 4.1: The proposed mobile robot exploration system architecture.

an *optimal proposal distribution* [86]. The RBPF-SLAM together with scan matching produces an *occupancy grid map* of an environment. Reactive navigation methods can be used to generate motion commands to navigate the robot to the target location. Here, we employ *nearness diagram* (ND) reactive navigation method proposed in [87]. However, the ND method is only applicable to circular shaped holonomic robots. In our experiments, we use a humanoid wheeled mobile robot which is non-circular and kinematically constrained. A *trajectory parameter space* (TP-Space) [88, 89] is used as an abstraction layer of the robot shape and kinematic constraints for ND method. We combine the TP-Space method with ND navigation approach for non-holonomic robot exploration in unknown environments.

The proposed integrated system for robot exploration in unknown environments aims to deal with imperfect control and sensing. In particular, this integrated approach does not require any distinguishable landmarks in the environment. It uses a RBPF to model the posterior about the trajectory of the vehicle using a finite set of particles. The

optimal proposal distribution can minimize the variance of the importance weights of the particles to reduce the uncertainty in the robot's world model over time. TP-Space based reactive navigation approach enables non-holonomic robots to navigate through their environment without colliding with obstacles. Using the proposed navigation target selection method, the robot can maximize its knowledge about the environment while avoiding obstacles close to it. In addition, the limitations of the *field-of-view* (FOV) of inexpensive depth sensors (such as Kinect sensor used here) are overcome by rotating the robot around the center of its wheel axle when it is necessary.

4.2 Optimal Particle Filter SLAM

In our approach for active SLAM, we use RBPF with an optimal proposal distribution for mapping with an occupancy grid map m. A complete derivation of this optimal particle filter for SLAM can be found in [86]. Let $x_{1:t} = x_1, x_2, \ldots, x_t$ be the robot trajectory which is obtained using the odometry measurements $u_{0:t} = u_0, u_1, \ldots, u_t$ and observations $z_{1:t} = z_1, z_2, \ldots, z_t$. RBPF is used to compute the posterior over maps and trajectories:

$$p(x_{1:t}, m|z_{1:t}, u_{0:t}) = p(m|x_{1:t}, z_{1:t}) \cdot p(x_{1:t}|z_{1:t}, u_{0:t}). \tag{4.1}$$

The posterior over maps $p(m|x_{1:t}, z_{1:t})$ are calculated when $x_{1:t}$ and $z_{1:t}$ are available. The posterior over the trajectories $p(x_{1:t}|z_{1:t}, u_{0:t})$ can be calculated sequentially by applying the Bayes rule:

$$p(x_{1:t}|z_{1:t}, u_{0:t}) = p(z_t|x_{1:t}, u_{0:t}) \cdot p(x_{1:t}|z_{1:t-1}, u_{0:t}). \tag{4.2}$$

In contrast to parametric models, the pose and map estimations obtained using the particle filter are represented using finite set of particles. A potential trajectory of the mobile robot is represented by each particle. Also a map hypothesis $m_t^{[i]}$ is associated with each sample. Let $\{x_t^{[i]}\}_{i=1}^N$ be a set of N robot path hypotheses at time step t,

distributed according to some proposal distribution:

$$x_t^{[i]} \sim \pi(x_t | x_{1:t-1}^{[i]}, z_{1:t}, u_{0:t}). \tag{4.3}$$

The variance of the importance weights $w^{[i]}$ can be minimized by selecting π based on the most recent observations [90]. This choice for π is referred as the optimal proposal distribution. According to Doucet *et al.* [90], the optimal proposal distribution that minimizes the variance of the next weights for any generic particle filter is given by,

$$\begin{aligned} \pi(x_t | x_{1:t-1}^{[i]}, z_{1:t}, u_{0:t}) &= p(x_t | x_{1:t-1}^{[i]}, z_{1:t}, u_{0:t}), \\ &= \frac{p(z_t | x_t, x_{1:t-1}^{[i]}, z_{1:t-1}, u_{0:t}) \cdot p(x_t | x_{t-1}^{[i]}, u_{0:t})}{p(z_t | x_{1:t-1}^{[i]}, z_{1:t-1}, u_{0:t})}. \end{aligned} \tag{4.4}$$

For the particles to represent the true posterior distribution, it is required to resample them at every time step. However, resampling at every time step may cause the particles, which represent the robot path, to lose its diversity. Here, *effective sample size* (ESS) [91] is used to measure the diversity of the particles. This quantity is computed as

$$ESS = \frac{1}{\sum_{i=0}^{N} (w^{[i]})^2} \tag{4.5}$$

The ESS is calculated at each iteration. If the ESS value is below the selected threshold, the particles are resampled using *rejection sampling*. The threshold for ESS is normally taken as half of the number of particles. In the optimal particle filter, a set of N particles is replicated into a set of auxiliary particles which are propagated according to the rejection sampling and the weights are updated according to the optimal proposal distribution [92]. In the resampling step, the final set of N samples are selected from the updated auxiliary particles with the probability proportional to their weights. This results in equal importance weights for all the selected particles.

In this work, we use occupancy grid maps as a non-parametric representation of the environment. The occupancy grid map consists of a set of grids, which are two dimensional in our case. Once the final set of particles are selected in the resampling step,

the corresponding map $m_t^{[i]}$ is estimated for each of those particles $x_t^{[i]}$ based on trajectory $x_{1:t}^{[i]}$ and observations $z_{1:t}$, according to $p(m_t^{[i]}|x_{1:t}^{[i]}, z_{1:t})$. The next target location is selected based on the estimated robot pose and up-to-date map.

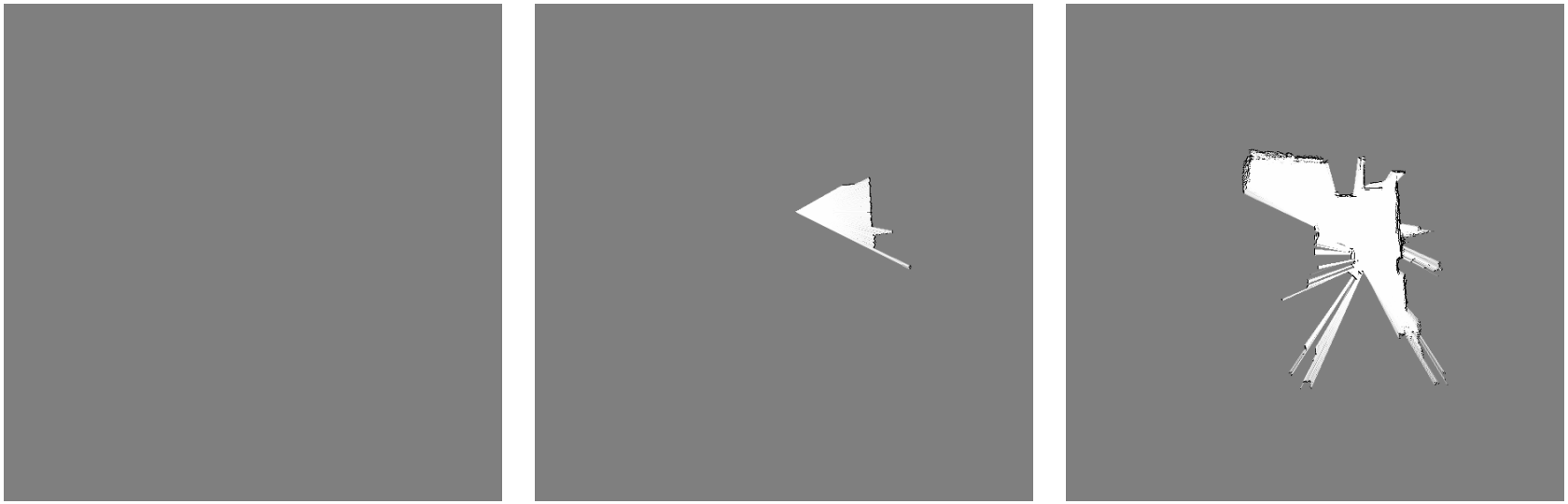
4.3 Navigation Target Selection

At the beginning of the exploration in a fully unknown environment, all the grids of the occupancy grid map are initialized with a prior probability value (normally 0.5). Figure 4.2(a) shows such an initial grid map. As the exploration process continues, the probability values of the grids reach 1 if those grids are occupied by obstacles (black pixels in Figure 4.2(b) and Figure 4.2(c) represents the obstacle boundaries), otherwise 0, if those are empty (white area in Figure 4.2(b) and Figure 4.2(c)). One of the main objectives of robot exploration is covering as much area as possible. Therefore the navigation targets should be selected so that the information is maximized.

4.3.1 Frontier-Based Exploration

One of the most popular approach for target selection is called as *frontier based exploration*, which was first proposed by Yamauchi [37]. Frontier cells define the boundary between explored and unexplored areas. These frontier cells offer the robot a possibility of visiting new places. If no frontier cells exist in the map, the robot has explored the total area under consideration and the navigation process will be stopped. The frontier grids are detected using techniques analogous to the detection and region extraction in computer vision. By moving the robot to new frontiers, it can extend its map to a new territory until the entire environment has been explored. The effect of false frontiers, generated mainly due to sensor errors, can be reduced by clustering the frontier grids and associating target candidates to the clusters with size comparable to robot dimensions.

As the map is updated according to $p(m_t^{[i]}|x_{1:t}^{[i]}, z_{1:t})$, new observations are inserted into



(a) (b) (c)

Figure 4.2: (a) An occupancy grid map of a fully unknown environment with prior probabilities, (b) An occupancy grid map with first range sensor scan which explores the area covered under its FOV, and (c) An occupancy grid map after several range sensor scans.

the map only when a new set of particles representing the robot trajectory is available. In the beginning there are no frontier grids in the map and the robot does not know where to navigate. If the robot initializes the navigating in any arbitrary direction, there is a chance of colliding with an obstacle as the map does not have any information at the beginning (Figure 4.2(a)). Also in partially explored environments, the robots might not be able to reach the frontiers due to its kinematic constraints.

4.3.2 Proposed Target Selection Method

With non-holonomic robots and range sensors with limited FOV ($< 90^o$), our target selection method borrows the ideas of frontiers with a more human-like approach. Imagine that you are looking for a specific room in a previously unknown building. How will you navigate inside the building without any assistance? You may look around and take a clear passage visible to you. Once you come to the end of the passage, you turn around and look for more free space to move. In our approach, we use similar idea for target selection. The proposed method is presented in the algorithm in Figure 4.3.

The algorithm starts with computing the clearance in front of the robot. In other words, the robot is looking for safe regions to navigate. In line 3, *compute_clearance* function returns the distance to the frontier grids or obstacles located straight ahead of the robot in the area covered under the FOV of the Kinect sensor. It also considers the width of the robot to determine whether the available free space is enough to occupy the robot or not. When the safety margin for the width of the robot is increased, its resistance to navigate in the narrow passages also increases, and vice versa. If the distance of clearance is greater than the *security distance* (d_s), it selects the next target position according to line 6 and line 7 in Figure 4.3. d_s is defined as the minimum distance between the center of the robot wheel axle and the obstacle boundary without any collision.

```
function SELECT_NEXT_TARGET(m_t, x_t, y_t, θ_t)
                                ▷ (x_t, y_t, θ_t) is the current robot pose, m_t is the map.
    clear_distance ← compute_clearance(m_t, x_t, y_t, θ_t)      ▷ returns the distance to
    frontier or to obstacle boundary.
    if clear_distance > min_allowed_distance then
                                ▷ Defines the coordinates of the navigation target.
        x_t^g ← x_t + clear_distance × cos(θ_t)
        y_t^g ← y_t + clear_distance × sin(θ_t)
        if nav_state_1 ≠ STRAIGHT then
            nav_state_2 ← nav_state_1
            nav_state_1 ← 0
        end if
    else if nav_state_1 ≠ STRAIGHT then
        (m_t, x_t, y_t, θ_t) ← rotate(m_t, nav_state_1, φ)
        nav_state_2 ← nav_state_1
        (x_t^g, y_t^g) ← select_next_target(m_t, x_t, y_t, θ_t)
    else
        (m_t, x_t, y_t, θ_t) ← rotate(m_t, −nav_state_2, φ)
        nav_state_1 ← −nav_state_2
        (x_t^g, y_t^g) ← select_next_target(m_t, x_t, y_t, θ_t)
    end if
    return (x_t^g, y_t^g)
end function
```

Figure 4.3: Navigation target selection algorithm. Function names are noted in *slanted_text*. Refer the Section 4.3.2 for more details.

nav_state_1 and nav_state_2 are system variables which carry the information about the navigation states of the robot in last two time states. Those variables are initialized and updated with three constant values $STRAIGHT$, $+ROTATE$, or $-ROTATE$ which correspond to three basic operations during the target selection:

- $STRAIGHT$ - moving straight,
- $+ROTATE$ - rotating counter clockwise around the center of the wheel axle, and
- $-ROTATE$ - rotating clockwise around the center of the wheel axle.

These two state variables are used in order to avoid robots being in circular loops. They are initialized with $\pm ROTATE$ and updated as the robot moves, in lines 9, 10, 14, and 18. If the distance of clearance calculated in line 3 is smaller than d_s, the robot cannot move forward anymore. Thus it rotates by an angle of ϕ either in clockwise or counter clockwise direction (line 13 and line 17). ϕ is a constant angle which is less than FOV of the range finder. The direction of rotation is decided according to the previous navigation states. If the operation command $+ROTATE$ or $-ROTATE$ is issued in the last time step ($nav\_state_1 = \pm ROTATE$), it will issue the same command in this time step as well. If the operation command $STRAIGHT$ is issued in the last time step ($nav\_state_1 = STRAIGHT$), it issues the opposite of the command issued in two time steps before ($-nav\_state_2$).

If we again consider the scenario of initializing the navigation in a fully unknown environment, using the proposed target selection method, the robot will first rotate an angle of ϕ as it does not have any prior information about the available safe regions, *i.e.* line 3 will return 0, which is obliviously less than d_s. However with the first rotation, robot explores some area of the map (as shown in Figure 4.2), which will let it decide the next navigation target. As the robot moves forward, it will keep exploring the area

in front. Therefore the distance of clearance returned in line 3 is not limited by frontiers as the maximum coverage distance of the range finder is greater than d_s in practice. Although the state variables hold the corresponding operations of the last two time steps in target selection process, it violates that when two $STRAIGHT$ commands are issued continuously. In such a scenario, it only updates nav_state_1 with $STRAIGHT$ ($nav\_state_1 = STRAIGHT$), and keeps the last rotational action in nav_state_2 (lines 8-10). The purpose of such a procedure is not to discard the information about the last rotational action.

Once the navigation target is selected in a global coordinate frame (x_t^g, y_t^g), it is combined with current robot pose to decide the navigation target in robot coordinate frame (x_t^r, y_t^r). Then the calculated relative target position is sent to the autonomous navigation system.

4.4 Reactive Navigation for Exploration

Autonomous navigation in an unknown and dynamic environment is one of the most challenging tasks in mobile robotics. This problem has been explored by mobile robot researchers for several decades. The mobile robot navigation algorithms can be classified into two broad categories: *motion planning* algorithms and *reactive navigation* algorithms. Motion planning algorithms compute collision free optimal path from the current robot position to the goal position for the known environmental models, *i.e.* when the map of the environment is given. Collision free mobile robot navigation in unknown, dynamic, and unstructured environments cannot be achieved through motion planning algorithms. As a solution, reactive navigation algorithms (*a.k.a.* real-time obstacle avoidance) have been proposed which periodically generates motion commands during real-time navigation directly from the sensory information.

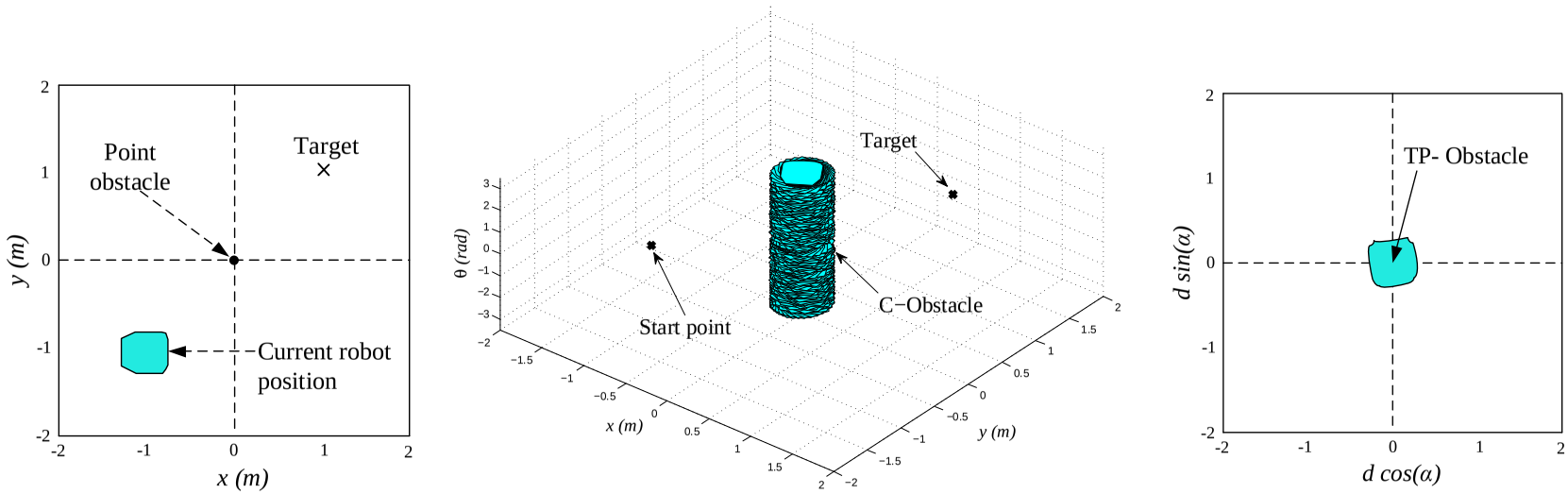


Figure 4.4: (a) An example workspace of a mobile robot with a point obstacle, (b) A C-Obstacle in the C-Space that results from the point obstacle in the workspace, and (c) A TP-Obstacle in the TP-Space, which can be seen as the intersection of the the sampling surface with the C-Obstacles.

4.4.1 Nearness Diagram Navigation

Most of the reactive navigation algorithms are highly computationally complex to use in real-time scenarios. The ND reactive navigation method could comprehensively reduce the computational complexity in real-time obstacle avoidance in very dense, untidily, and complex environments. The ND method simplifies the navigation problem based on a divide-and-conquer strategy that defines a set of complete and exclusive situations. Once the situations are identified from the range data, the corresponding actions are applied which address the relative state of each problem entities. This perception-action process is completed using a situated-activity paradigm of the behavioral design.

The reactive navigation system based on the situated-activity paradigm periodically receives the Kinect sensor data, relative goal location and estimated robot location. It analyzes the current safety level of the robot using those information. If there are obstacles close to the robot, it reports that the robot is in *low safety*, otherwise in *high safety*. The high safety situation is further divided into sub-situations according to the width of the passage connecting the robot and current goal location. If the goal is within the safety region in front of the robot, it directs the robot towards the goal. If the robot is facing a wide safety region and the goal is not within that area, it drives the robot close to the obstacle till it passes it. If the safety region is narrow and the robot can move without any collisions, it drives the robot through the central zone of the safety region. The low safety situation is also divided into sub-situations. If there are obstacles only on one side of the gap closest to the goal, it first moves the robot away from the closest obstacle and then towards the safety region. If there are obstacles on both the sides, it centers the robot between the obstacles and drives the robot towards the safety region.

The motion command of the robot is also decided using the situated-activity paradigm of the behavioral design. If the robot is in a high safety situation in the tth time step,

the absolute value of the linear velocity is defined as

$$\tilde{v}_t = \tilde{v}_{max} \frac{(\frac{\beta}{2} - |\gamma|)}{(\frac{\beta}{2})}, \tag{4.6}$$

where $\tilde{v}_{max}$ is the maximum linear velocity, β is the FOV of the sensor, and γ is the linear velocity direction. The robot moves at the maximum speed when there is no obstacles ahead of it and $\gamma = 0$. It reduces the speed as it sees an obstacle. When it reaches low safety regions, the linear velocity is updated as

$$\tilde{v}_t = \tilde{v}_{max} \frac{d_{obs}}{d_s} \frac{(\frac{\beta}{2} - |\gamma|)}{(\frac{\beta}{2})}, \tag{4.7}$$

where d_{obs} is the distance from the closest obstacle to the robot bounds. It reduces $\tilde{v}_t$ as it gets close to the obstacles. According to Equations (4.6) and (4.7), the absolute value of the linear velocity is also reduced as γ increases. Since we use a depth sensor with limited FOV which has been attached to the front of the robot, the robot avoid instantaneous backward motion. Therefore, direction of the linear velocity is restricted to $\gamma \in [-\beta/2, \beta/2]$. In both the low safety and high safety situations, the angular velocity is defined as

$$\tilde{\omega}_t = \tilde{\omega}_{max} \frac{\gamma}{(\frac{\beta}{2})}, \tag{4.8}$$

where $\tilde{\omega}_{max}$ is the maximum angular velocity.

Although the ND method is a less computationally complex solution for obstacle avoidance in troublesome scenarios, it can only be employed with circular shaped holonomic robots. Therefore, we need to abstract vehicle shape and kinematic constraints before using ND navigation method as illustrated in Figure 4.5.

4.4.2 Trajectory Parameter Space

TP-Space is used as an abstraction layer of the robot shape and kinematic constraints for ND method. In TP-Space, problems of kinematics restrictions and obstacle avoidance

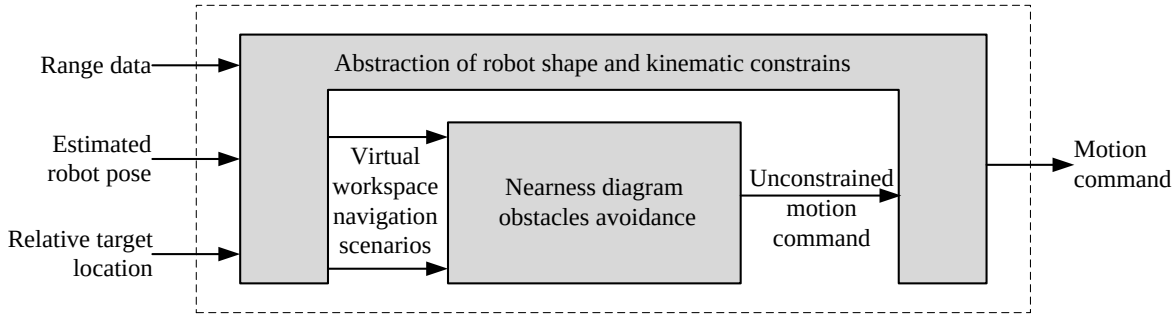


Figure 4.5: TP-Space based reactive navigation system.

can be separated by using path models to transform compatible paths and workspace obstacles into a lower dimensional space. Therefore the robot can be treated as a free-flying-object in TP-Space and ND navigation can be used for obstacle avoidance. Figure 4.4 illustrates the conversion process from workspace to TP-Space. In motion planning approaches, *configuration space* (C-Space) is used to model the environment [93]. A mobile robot navigating in planar environment is represented by using three dimensions in C-Space: x,y, and θ. In the occupancy grids maps, the obstacles are represented as a combination of several points (occupied grids). An example workspace where a mobile robot navigating in a planar environment with a point obstacle is shown in Figure 4.4(a). The robot is represented as a point in the C-Space. The missing information about the robot shape is carried by the obstacles in the C-Space (called as C-Obstacles). However the C-Obstacles assume robots to be holonomic which is not always true in practice. In C-Space motion planning, any 3D curve linking the start pose and target pose which does not go through the C-Obstacle, can be selected as a continuous sequence of robot poses to reach the goal. However, in practice, not all of those pose sequences can be realized with non-holonomic mobile robots.

In order to abstract the kinematic constraints, we extract the 3D curves from the C-Space which satisfy the path models introduced in [89]. If we visualize all such possible paths from a given model, we obtain a 3D surface called as a *sampling surface*. TP-

Space is obtained by straightening out the sampling surface. An example for such a straightened out sampling space (TP-Space) is shown in Figure 4.4(c). Therefore, each point in TP-Space corresponds to a pose within a C-Space sampling surface. TP-Space is normally represented in polar coordinates: an angular component α and a distance d. The mapping between a TP-Space and a sampling surface is achieved by selecting a trajectory from a specific path model using α. The distance of the pose along the selected trajectory is substantiated by d. However, once the mapping is done, obstacle avoidance is achieved by the ND approach. However the motion command $(\tilde{v}_t, \tilde{\omega}_t)$ decided by ND method is only valid in the virtual TP-Space and needs to be mapped back to the real workspace motion command (v_t, ω_t) using the transformations explained in [89].

4.5 Experiment Results

The proposed reactive navigation based exploration technique is tested in indoor environments with the H20 humanoid wheeled mobile robot (WMR) shown in Figure 2.2(a). The H20 is a non-holonomic mobile robot with a non-circular shape. The shape of the base of the mobile robot platform is shown in Figure 2.2(b). A detailed discussion on H20 WMR is provided in Section 2.1.3. In this work, we obtain the distance to the obstacles in the indoor environments using a Kinect sensor [63] mounted on H20. To the best of the our knowledge, this Kinect sensor has not been used in robot exploration application, however it has been used in indoor robot localization in [61]. The optimal particle filter SLAM algorithm and TP-Space based reactive navigation algorithm is implemented using the mobile robot programming toolkit [94]

We present here two experiments carried out using H20 mobile robot in different parts of our laboratory at the University of Calgary. The environment was fully unknown to the robot at the beginning of each experiment. The robot started exploration in a random position in the test environment with any arbitrary orientation. The results

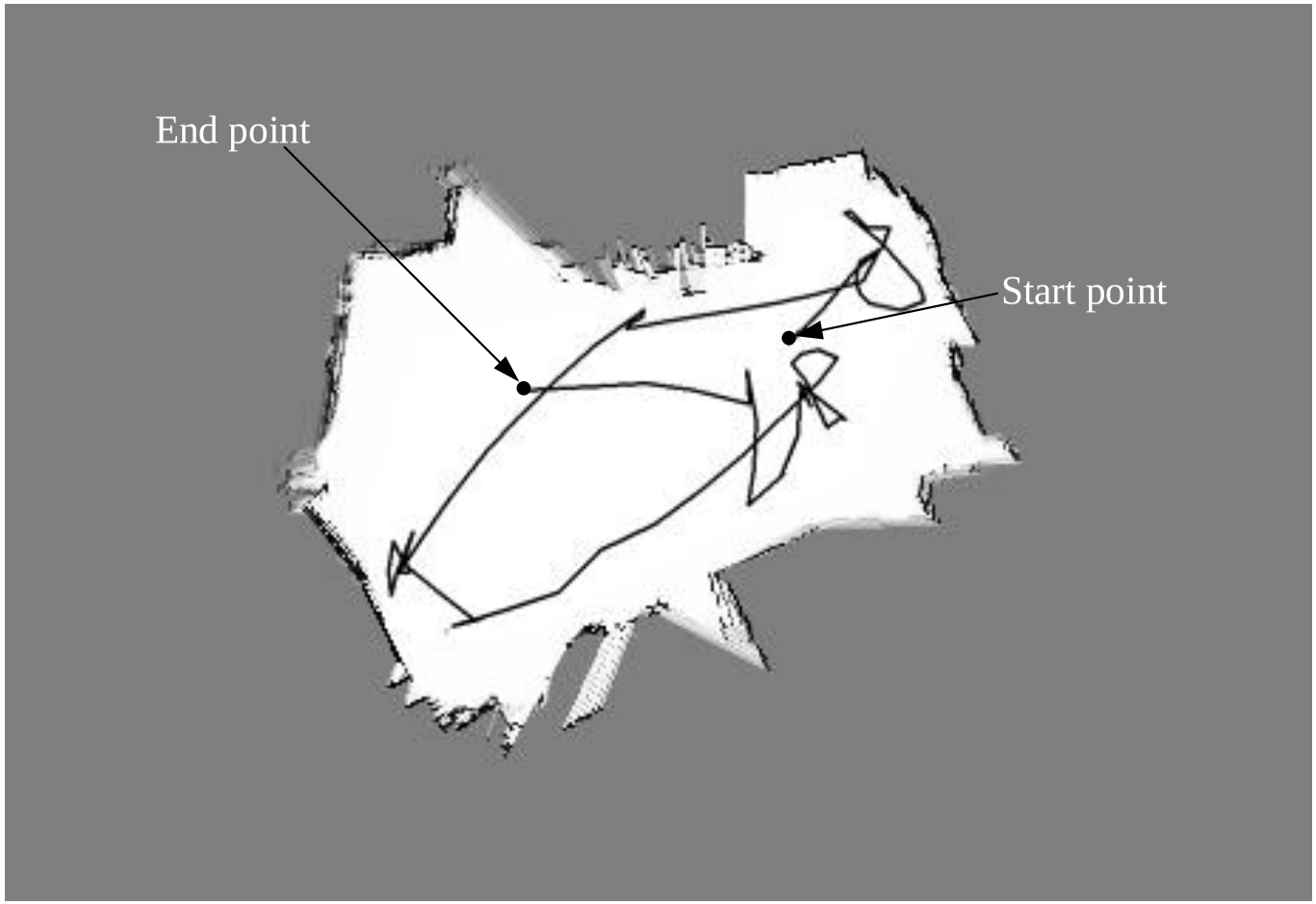
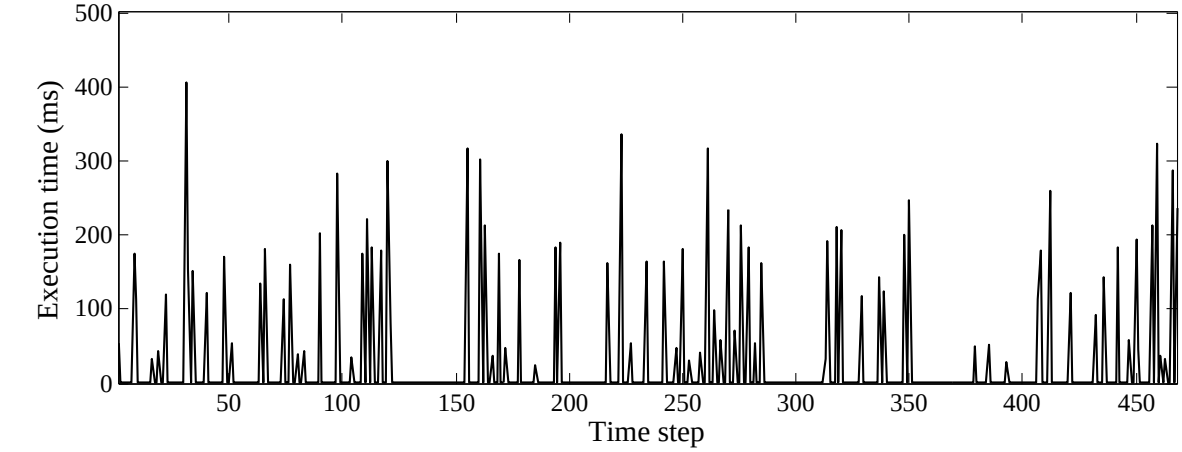
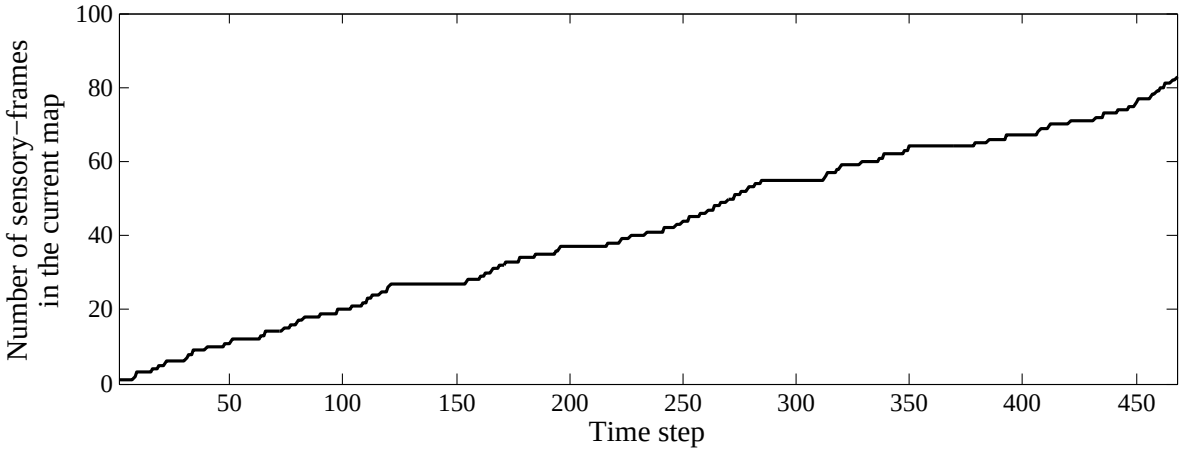


Figure 4.6: Results of the first experiment: Occupancy grid map of the environment with the estimated robot trajectory.

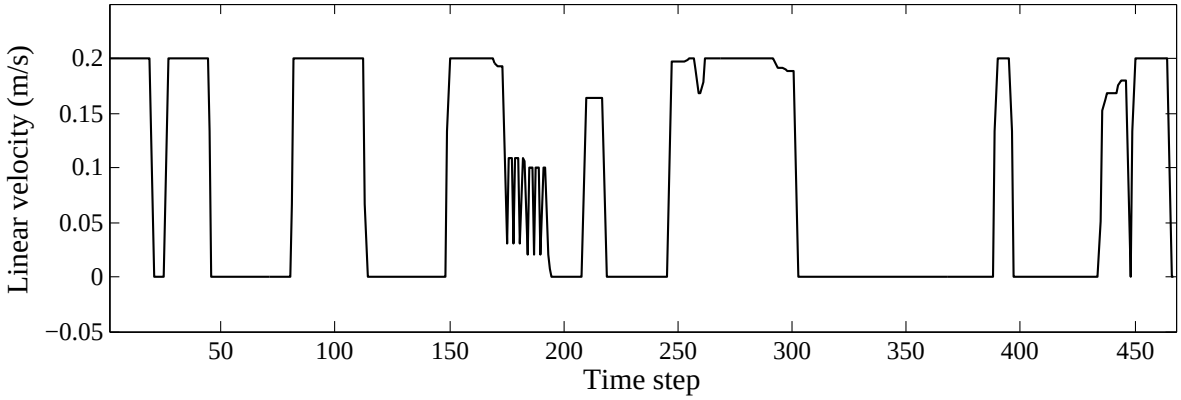


(a)

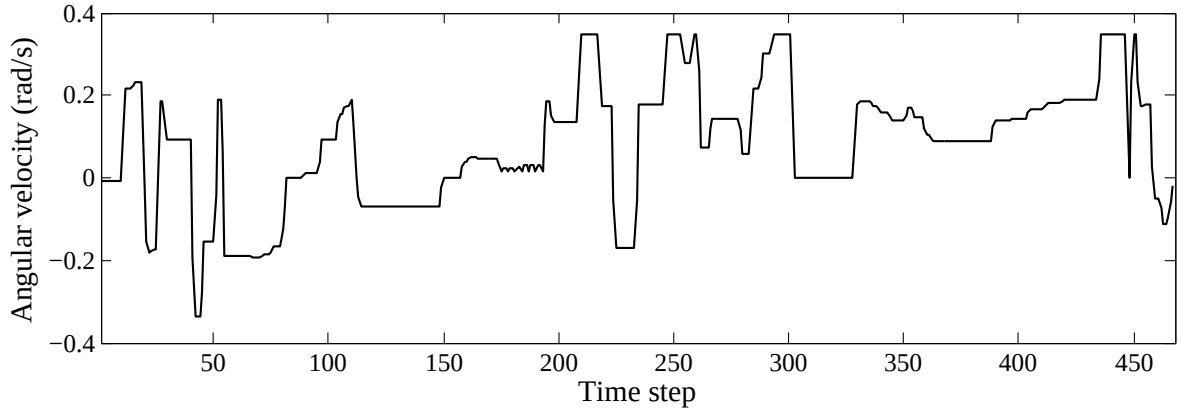


(b)

Figure 4.7: Results of the first experiment: (a) Execution time of the map building process and (b) Number of sensory-frames in the current map.



(a)



(b)

Figure 4.8: Results of the first experiment: (a) Linear velocity of the robot and (b) Angular velocity of the robot.

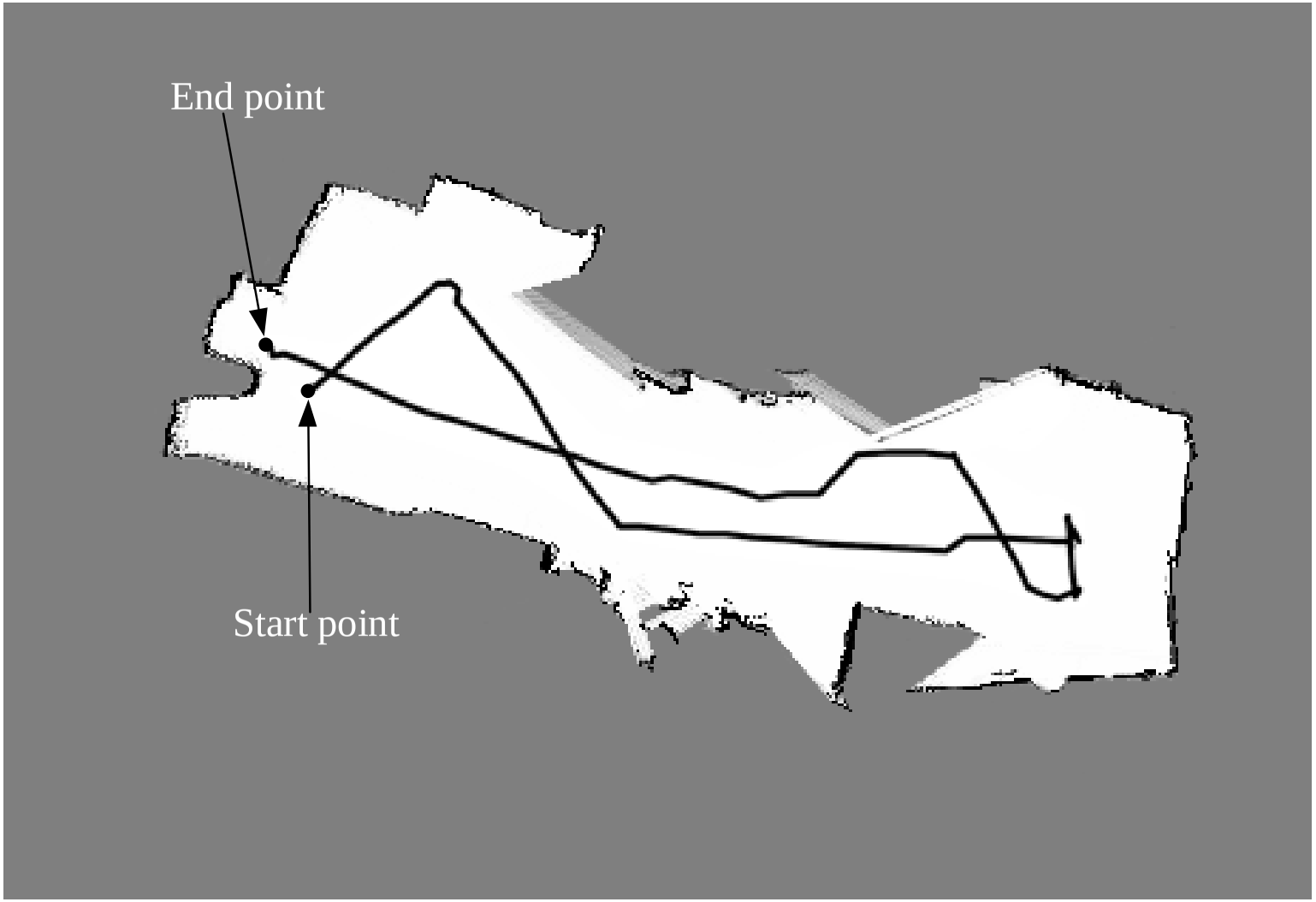
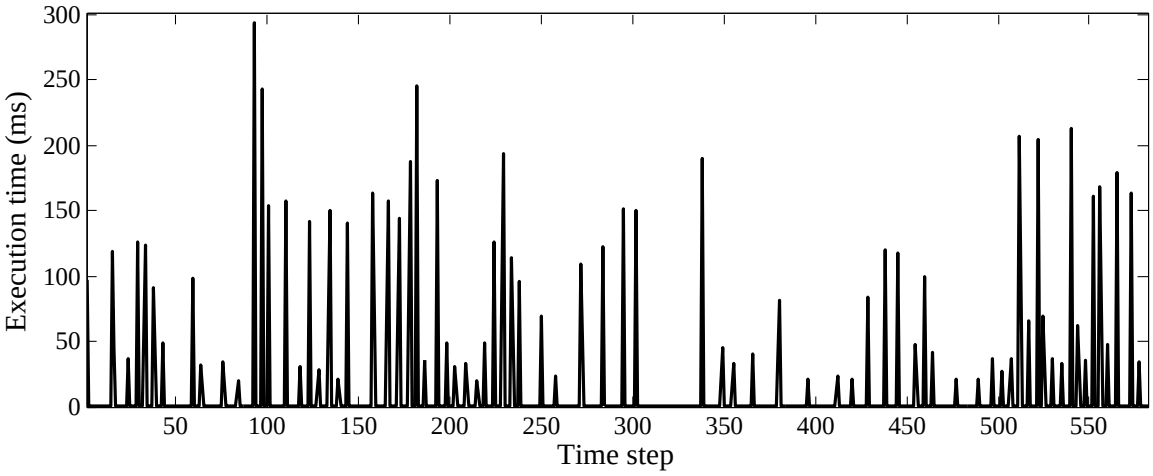
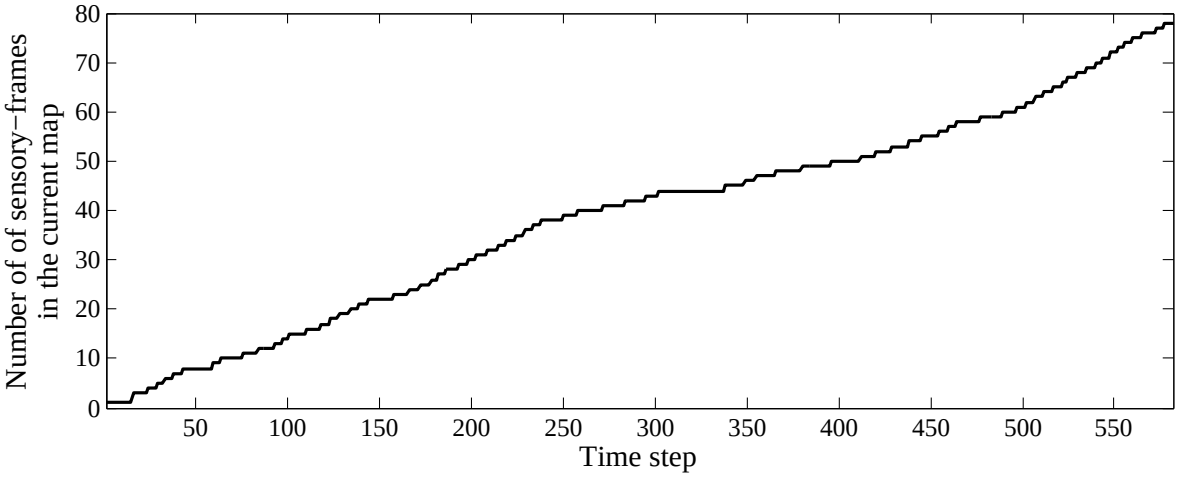


Figure 4.9: Results of the second experiment: Occupancy grid map of the environment with the estimated robot trajectory.

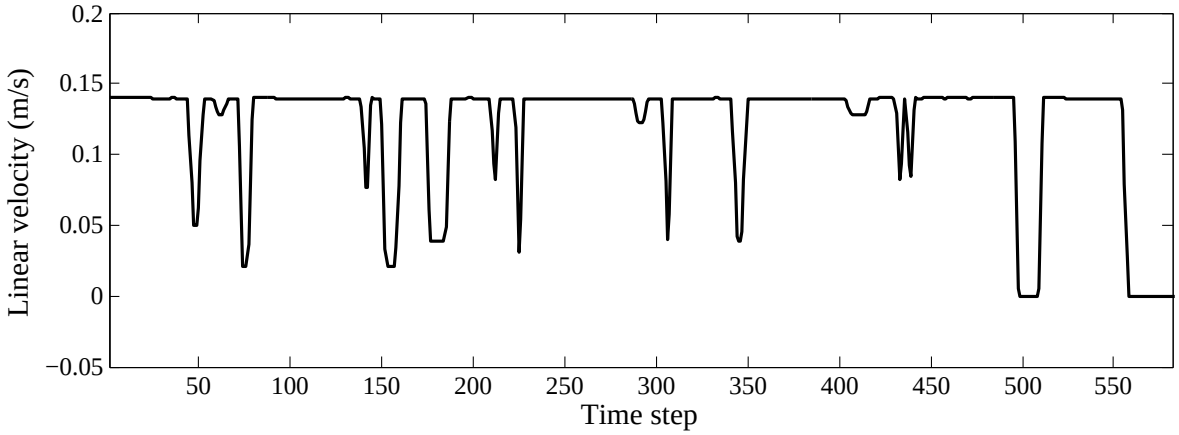


(a)

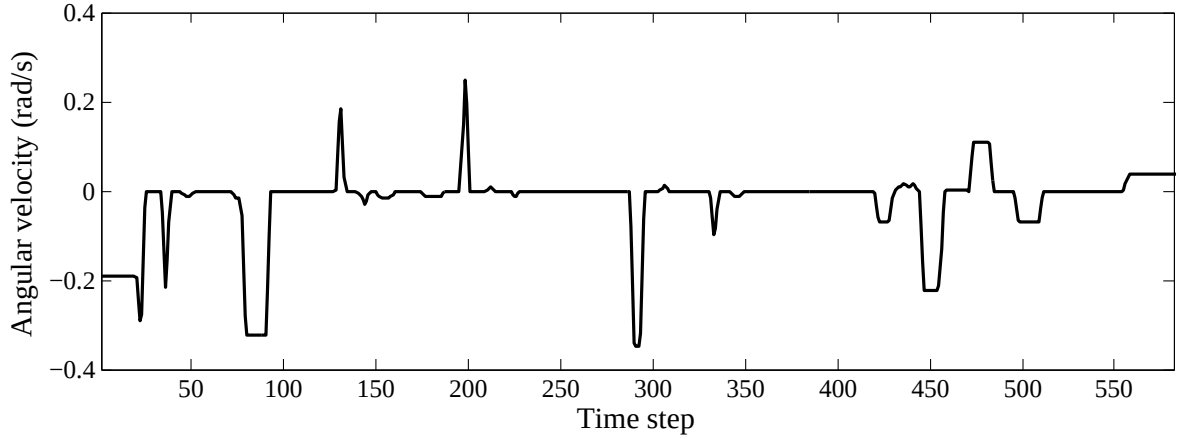


(b)

Figure 4.10: Results of the second experiment: (a) Execution time of the map building process and (b) Number of sensory-frames in the current map.



(a)



(b)

Figure 4.11: Results of the second experiment: (a) Linear velocity of the robot and (b) Angular velocity of the robot.

of the experiments are presented in Figures 4.6 – 4.11, in order to verify the three goals of mobile robot exploration: map building, localization and motion control. The experiments were carried out without any kind of human interaction, except for issuing the commands to start and conclude the exploration.

In the first experiment, the robot had to explore an unstructured passage with hard asymmetries. We set the maximum linear velocity to 0.2 m/s and maximum rotational velocity to 0.34 rad/s (20 deg/s). The occupancy grid map of the test environment and estimated robot trajectory are shown in the Figure 4.6. The map was built using the RBPF based on 83 sensory-frames from the Kinect sensor shown in Figure 4.7 (b). The execution time of each map building step is shown in Figure 4.7 (a). The motion commands were generated using the ND approach based on the Kinect sensor measurements. Figure 4.8 (a) and Figure 4.8 (b) illustrate the linear and angular velocity components of the motion commands in each time step. During its complete exploration, the robot navigated 12.35 m without any collisions. In the second experiment, the robot also had to explore an unknown and unstructured environment, which was longer compared to the previous one. Here, we set the maximum linear velocity to 0.14 m/s and maximum rotational velocity to 0.34 rad/s (20 deg/s). The robot navigated 14.74 m without any collisions throughout the whole run. The results of the second experiment are shown in Figures 4.9 – 4.11.

4.6 Summary

Autonomous exploration is a key component in search and rescue robots. Simultaneous localization, mapping, and motion controlling are required for exploring unknown environments with such robots. Although the SLAM problem is widely discussed among the robotic research community, the problem of simultaneous motion controlling in an unknown environment has been unable to attract the adequate attention. In this work,

we address the mobile robot exploration problem using an integrated system which is capable of generating motion commands for the robot while performing the SLAM. We do not make any assumptions about distinguishable landmarks in the environment with the occupancy grid maps used in this application. The optimal particle filter SLAM models the posterior about the robot trajectory while minimizing the variance of the importance weights of the particles. We use a human-like approach to select the navigation target in the up-to-date map. The ND reactive navigation system generates the motion commands based on the estimated robot position and next target position. As we carried out our experiments with a non-circular and non-holonomic mobile robot, we used TP-Space as an abstraction layer of the robot shape and kinematic constraints for ND navigation method. The proposed system was implemented and tested on a mobile robot platform with an inexpensive sensor system. Despite its low cost sensor system, experimental results have validated the robustness and real-time applicability of the proposed method.

Chapter 5

High Level Data Fusion for Goal Detection

5.1 Introduction

In many applications, mobile robots have to decide the location of their goal for motion planning. These goals depend on the application that the robots are utilized in. For example, the mobile robots utilized in military battlefields need to determine their enemies and allies correctly based on their behavior and appearance. Search and rescue robots need to identify the victims in hazardous environments to assist human workers. Such real-time applications require on-line fusion of sensory data so that the *situation awareness* is maximized. Within a given volume of time and space, situation awareness can be defined as the perception of the elements in the environment, comparison of their meaning and projection of their status in near future [95]. This chapter proposes a system developed with the objective of reaching targets under tangible situations based on sensor information.

State-of-the-art mobile robot platforms use multiple sensors to gather information about their environment. Those sensors may be used to observe a single situation in the environment. The proposed *situation assessment framework* (SAF) analyzes sensory information with respect to prior knowledge to provide decision support. A *fuzzy cognitive map* (FCM) based SAF has been previously used in CanCoastWatch project for high level data fusion [96]. Situation assessment is also used in unmanned air vehicles to organize and represent a skilled human's situation assessment behavior in troublesome situations [97]. The concept of FCM is introduced in [98, 99] by considering fuzzy values for the concepts in the cognitive maps. FCMs have been applied in many different

applications such as analyzing electrical circuits [100], modeling intelligent supervisory control systems [101], and analyzing graph theoretical behavior [102]. The proposed FCM based SAF for navigation goal detection can efficiently process the information coming from a large number of sensors and make the decisions effectively.

An experimental scenario is developed as a proof of concept and therefore has limited number of situations. Here, the navigation goal of the robot is to find a cup on a box in an office environment. The robot needs to observe the environment using its sensors and identify the exact situation which matches its goal, *i.e.* the proposed system should be able to distinguish between isolated boxes and a box with a cup on top of it as its goal location. Here, we use Microsoft Kinect as our sensing device [63]. The RGB camera and depth sensor of Kinect are used as two separate input devices to our decision fusion architecture. The information captured by RGB camera is used to observe the color of the box and the cup. The data received through the depth sensor of the Kinect is used for recognizing the shape of the objects.

5.2 Situation Assessment Framework

Situation assessment has been used in different domains [96, 103, 97]. In the context of information fusion, it can be defined as a process of estimating and predicting the relationship among entities while including the physical context, perceptual influences, and communications [104]. The hierarchy of components in a data fusion model is given in Table 5.1 [105]. The first two steps of this data fusion model can be recognized as feature extraction and object detection. Situation assessment comes third in the data fusion hierarchy. In our case, recognizing the color or shape of a box or a cup can be categorized under feature extraction, which uses relevant features from sensor readings such as RGB (red, green, and blue) data and depth information. Recognizing the box or cup can be identified as object detection. Somehow, in order to evaluate most real-world

Table 5.1: Hierarchy of components of a data fusion system.

Data fusion level	**Estimation process**	**Entity estimates**
L0: Sub-object assessment	Detection	Signal
L1: Object assessment	Attribution	Physical Object
L2: Situation assessment	Relation	Situation
L3: Impact assessment	Plan interaction	Effect
L4: Process refinement	Control	Action

scenarios, we actually need to assess the situation. For example, if you are asked to find a coffee mug, you will not look for it inside drawers, instead, you will look for it on the tables or in cupboards. Searching in the most likely cup locations first, reduces the time needed for locating a cup. Similarly in our work, we have defined our target test case as locating a cup, which the robot will start looking for on tables (instead of tables, we use boxes in our experiments). Situation assessment is highly useful in such scenarios as it combines expert knowledge and sensor information together.

The primary objective of this work is to develop a goal-driven SAF wherein a navigation goal needs to be verified using prior knowledge and sensor information. The proposed SAF for navigation goal detection is shown in Figure 5.1. The robot's navigation goal is passed to the SAF and the orientation manager describes it through appropriate sub-goals. The degeneration of a goal will continue until the sensor readings are sufficient to describe the sub-goals. The break down of various goals into sub-goals is defined by the expert knowledge. Taking our setup as an example, the goal is to detect a cup located on a box (*isCupOnBox*). This is defined using sub-goals that can be identified as intermediate decisions, such as *isCorrectBox* and *isCorrectCup*. The proposed goal-driven situation assessment works backwards from the initial goal and tries to prove it by associating it to several sub-goals.

The inference mechanism in SAF first selects the *rules* with conclusions matching the goal. Rules are stored in *knowledgebase*, for example: IF *isCorrectBox* AND *isCorrectCup* THEN *isCupOnBox*. Likewise, multiple rules can be stored within the

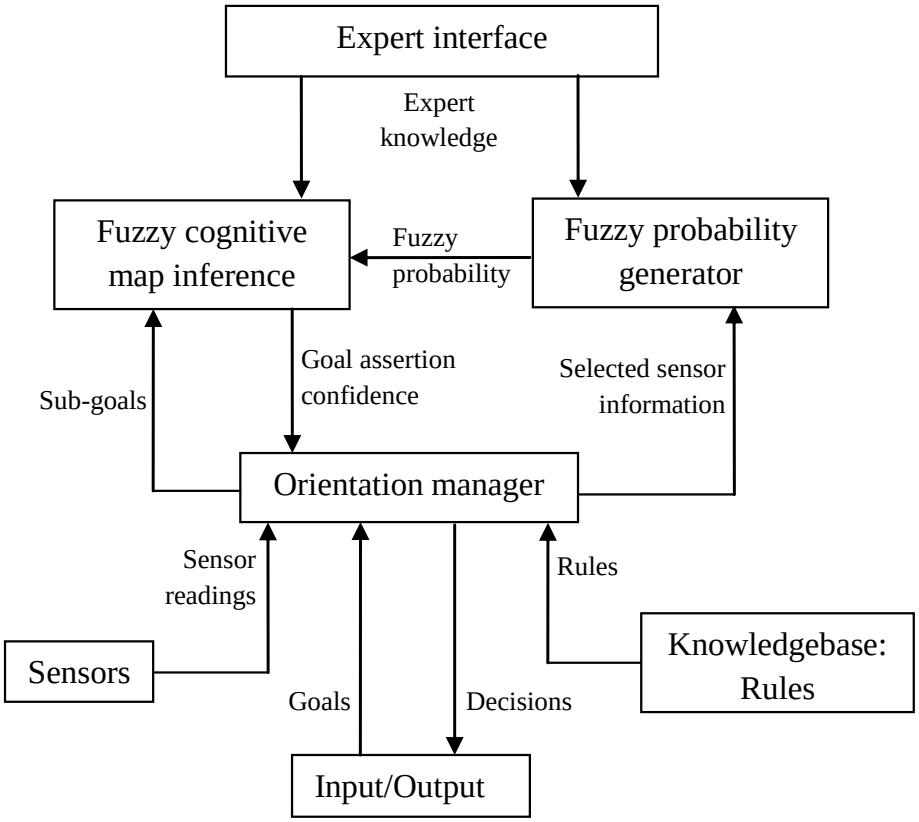


Figure 5.1: Structure for situation assessment.

knowledgebase depending upon the number of goals needed to be verified by the SAF. Once the correct rule is extracted from the knowledgebase corresponding to its currently assigned goal, the goal is replaced by the rule's premises which then become sub-goals. Then it passes the information about sub-goals to the FCM. Verification of each sub-goal is achieved by the FCM using the fuzzy probability and prior knowledge. Goal assertion confidence for each sub-goal is decided using FCM inference and returned to the orientation manager. The SAF recursively works backward until all sub-goals are verified. Consideration of the next sub-goal depends upon the rules and assertion confidence of previous sub-goals. If we consider the previously mentioned rule as an example, a failure in assertion of either sub-goal will result in a negative outcome of the final decision as all the sub-goals are connected using AND operators.

5.3 High Level Data Fusion using Fuzzy Cognitive Maps

FCM can be considered as a combination of fuzzy logic and neural networks [106]. The proposed system occupies multiple FCMs for high level decision making in detecting navigation goals. Once sub-goals have been identified, the fuzzy cognitive map inference uses fuzzy probabilities and expert knowledge to assert goal confidence. Fuzzy probabilities (membership values) are generated by the *fuzzy probability generator* (FPG) which relies on selected sensor information and expert knowledge in the form of fuzzy membership functions for color and shape of objects. Fuzzy membership functions for color and shape of boxes are shown in Figure 5.2. Similar membership functions can be obtained for the cup as well. Based on the degree of match in color and shape, described in Section 5.3.2, fuzzy probabilities are generated for all selected sensor information. The expert knowledge within the FCM inference refers to the method of fuzzy probability fusion. Here, fuzzy gamma fusion is used to combine the fuzzy probabilities generated by the FPG.

5.3.1 Fuzzy Gamma Fusion

Given multiple fuzzy membership functions for the same situation, there are several operators that can be used to combine membership values together. Fuzzy gamma fusion is one such method which incorporates expert knowledge for combining data sets [107]. It can be defined as

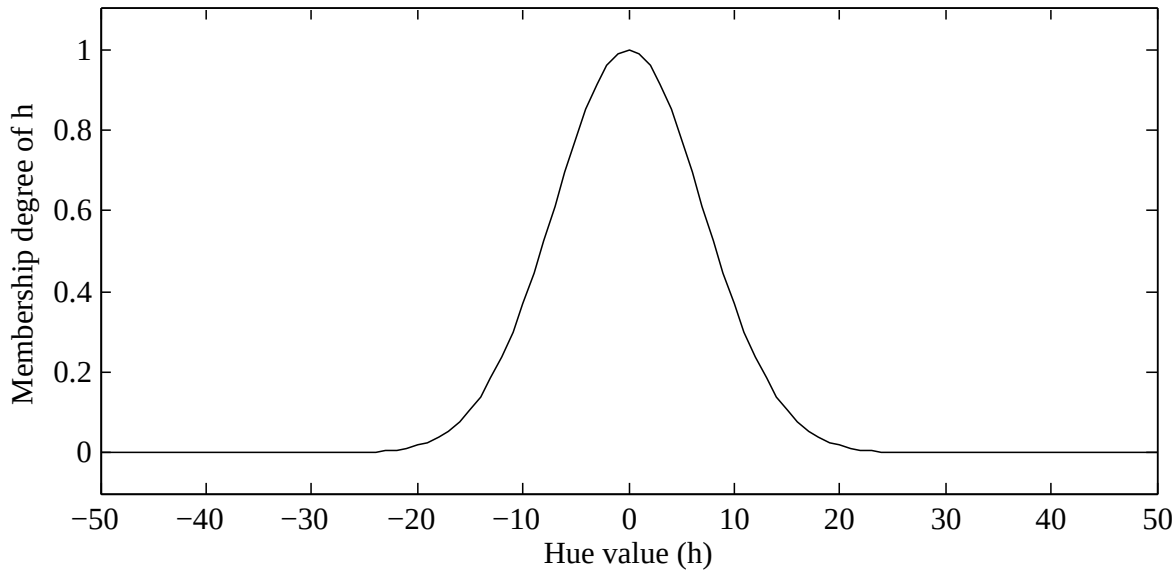
$$\mu_{fused}(x) = \left(1 - \prod_{i=1}^{n}(1 - \mu_i(x))\right)^{\gamma} \times \left(\prod_{i=1}^{n}\mu_i(x)\right)^{(1-\gamma)} \tag{5.1}$$

where μ_i is the membership value of ith membership function and μ_{fused} is the assertion confidence for each sub-goal generated by FCM inference. Fuzzy gamma fusion provides a flexible compromise between the *increasive* effects of the fuzzy algebraic sum and *decreasive* effects of the fuzzy algebraic product where γ is judiciously given by the

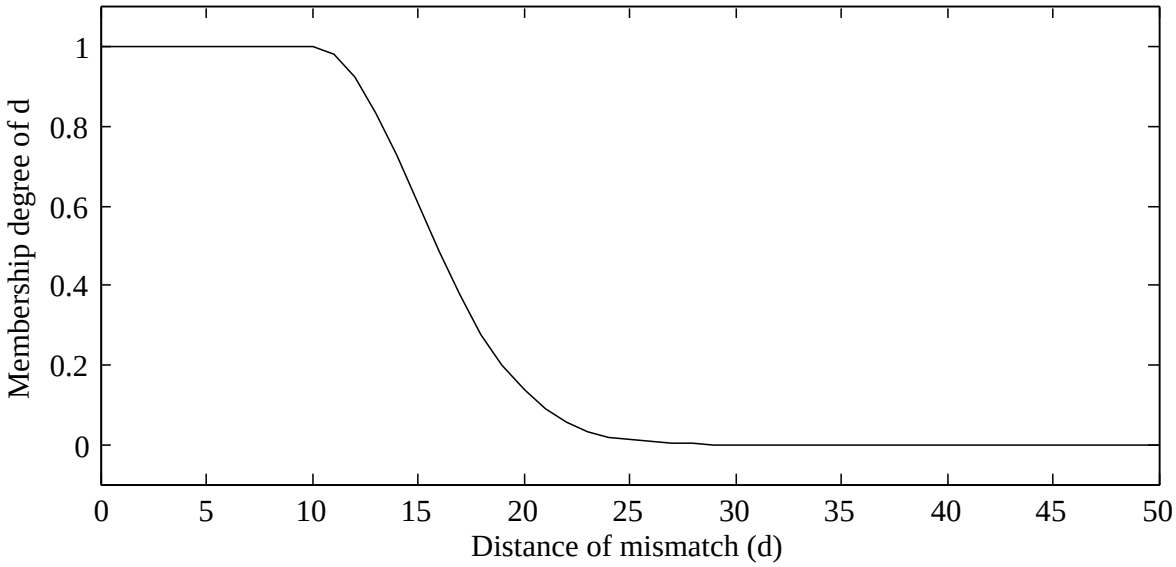
expert interface to reflect the subjective decision-making of a typical human. By varying γ from 0 to 1, fuzzy gamma fusion changes from a purely fuzzy algebraic product to a purely fuzzy algebraic sum. Although, the fuzzy algebraic sum provides the largest maximization to object detection, it is susceptible to false positives within feature fusion. For example, $\mu_{color}(x) = 0.1$ and $\mu_{shape}(x) = 1.0$ corresponds to a perfect shape but a poor color match. Therefore, when $\gamma = 1$, $\mu_{fused}(x) = 1.0$ and an incorrect assertion is made by the FCM inference resulting in a false positive generated by the orientation manager. However, if $\gamma = 0.7$, $\mu_{fused}(x) = 0.50$ is sent to the orientation manager which can refer to its knowledgebase rules for detection thresholds. Figure 5.3 illustrates the effect of different γ values on detection probabilities.

5.3.2 Feature Extraction for Object Recognition

For real-time situation assessment, the 3D shape recognition must be fast and reliable. Kinect provides dense depth and RGB maps, here referred to as point clouds. However, the raw point cloud data is dense and comes with high computational cost, therefore filtering and segmentation are required before recognition can be carried out by the fuzzy probability generator. The 3D shape recognition module was implemented using *Point Cloud Library* (PCL) [108]. The raw point cloud is first down-sampled using a 3D voxelized grid filter and then segmented using Euclidean cluster extraction [109]. The clusters are then transformed into a *viewpoint feature histogram* (VFH), an accurate, computationally efficient 3D feature descriptor [110] and used in a fast, simple nearest neighbor (NN) classifier in [111]. The NN classifier provides the degree of match between an arbitrary VFH and a trained VFH as a distance measurement (which is referred to as distance of mismatch in Figure 5.2 (b)), where 0 corresponds to an exact match. The orientation manager sends distance of mismatch as the selected sensor information to the FPG for probability generation.

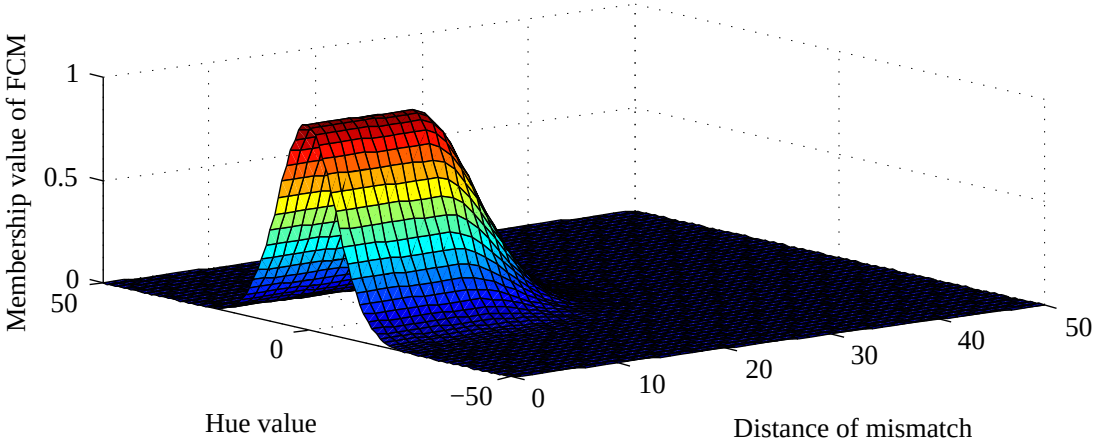


(a)

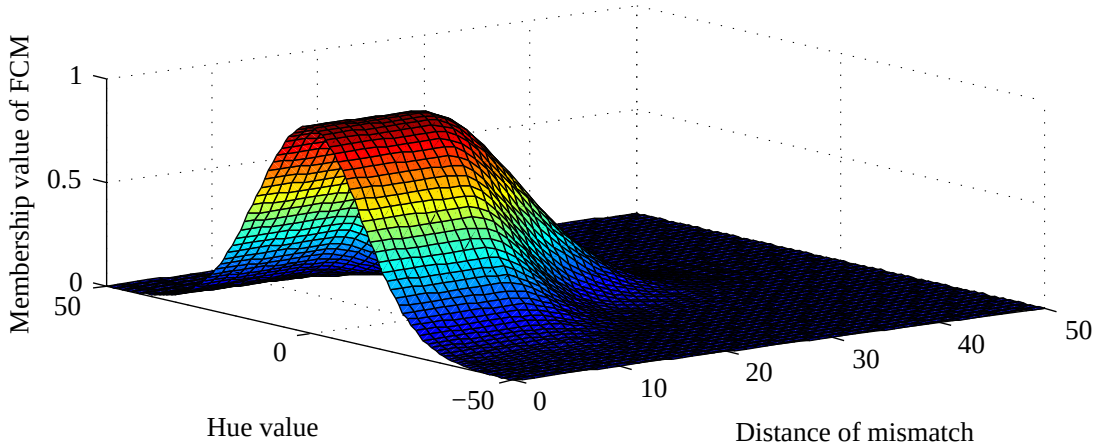


(b)

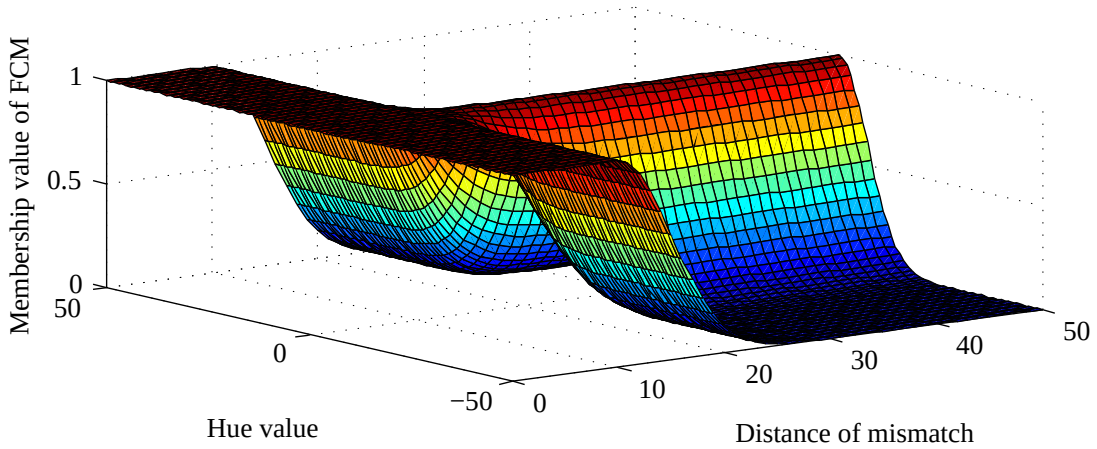
Figure 5.2: Membership functions for the box: (a) membership degree of hue value (μ_{color}) and (b) membership degree of distance of mismatch (μ_{shape}).



(a)



(b)



(c)

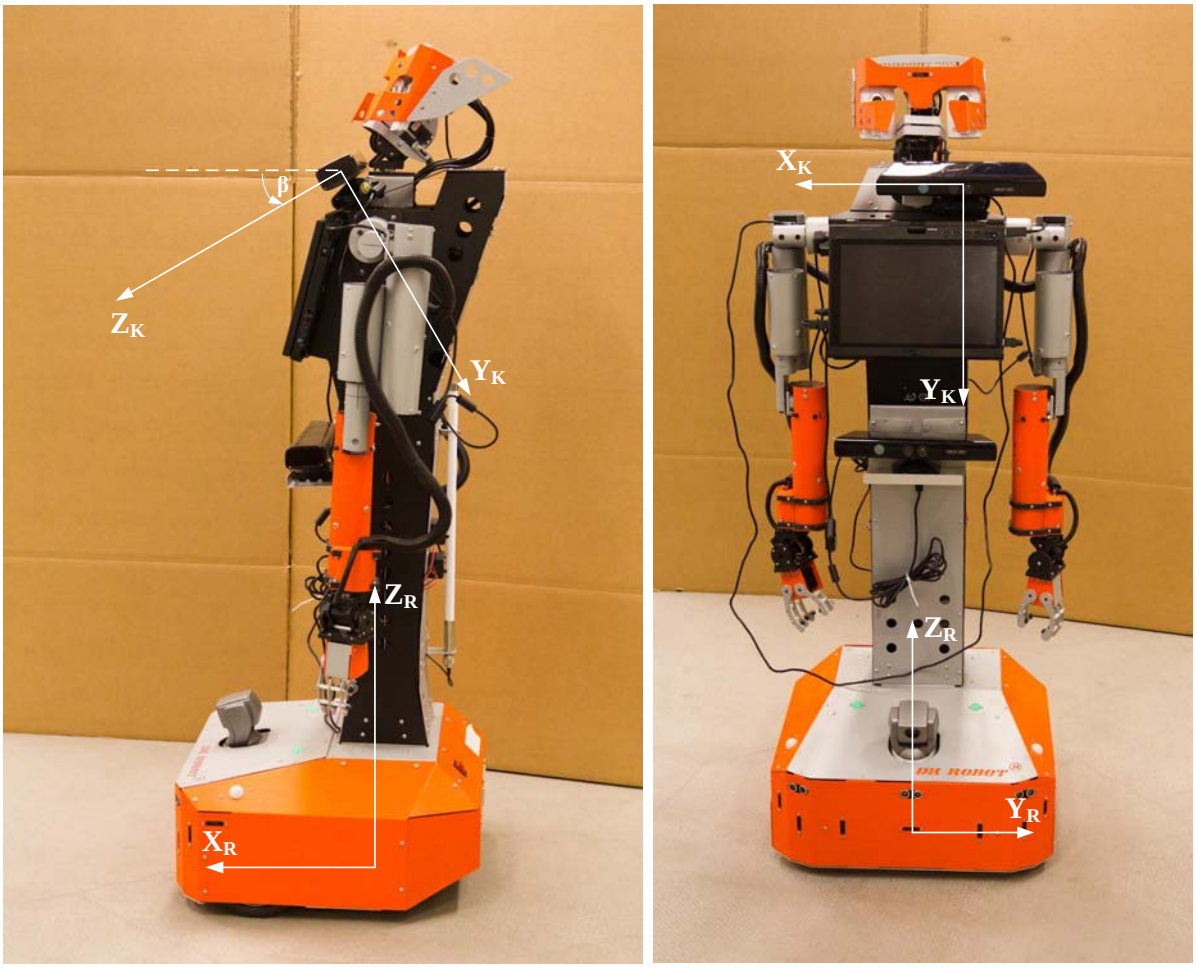
Figure 5.3: Membership values of FCM for *isCorrectBox* (a) when $\gamma = 0$, (b) when $\gamma = 0.7$, and (c) when $\gamma = 1$.

Since the VFH descriptor only makes distinctions based on 3D features, additional features were required to differentiate between geometrically similar objects. RGB color was an easily extracted feature from the point cloud data but the additive blend of RGB values that make up an arbitrary color are unintuitive and are highly prone to variation with changing lighting conditions. Therefore, the RGB colors were transformed into hue, saturation, and value (HSV) and the averaged hue was used as the color feature of an arbitrary point cloud. The HSV color model is more robust to changing light conditions which typically affect the saturation and value in HSV descriptions. The degree of match between an arbitrary hue and a trained hue is modeled as a Gaussian function with a mean defined by the average trained hue and variance defined by the correlation between training sample hues (*see* Figure 5.2(a)). The orientation manager sends the average hue value of clusters as selected sensor information to the FPG for probability generation.

5.4 Goal Location Estimation

After the SAF correctly identifies the goal, its location has to be decided by the robot in order for it to reach there. In this implementation, a dedicated Kinect sensor is used for goal detection in addition to the Kinect sensor which is used for obstacle avoidance. The H20 mobile robot with both the Kinect sensors is shown in Figure 5.4. The upper Kinect is used for goal detection. As explained in Section 2.2.3, the Kinect sensor has the problem of close range blind spot. The appropriate position and attachment of the second Kinect sensor renders the robot with a better capability of seeing nearby objects.

Unfortunately it is not possible to get a direct measurement of the goal location with respect to the same coordinate frame where the robot is localized. Instead the PCL gives the location vector of the goal with respect to the Kinect coordinate frame (X_K, Y_K, Z_K). Also the coordinate frames of the robot (X_R, Y_R, Z_R) and the Kinect sensor are not



(a) (b)

Figure 5.4: H20 mobile robot with a dedicated Kinect sensor for goal detection. (a) Side view and (b) front view of the robot which illustrates the relative position and orientation of the the local coordinate frames of the Kinect and robot.

aligned with each other. Figure 5.4 illustrates front and side views of the H20 robot with the position and orientation of each coordinate frames. The transformation of these coordinates to the robot's coordinate frame can be accomplished using the homogeneous transformations which combine the position vectors and rotation matrices into a compact notation [58]. Let ${}^{r}\mathbf{T}_k$ be the transformation matrix from Kinect coordinate frame to robot coordinate frame which is given by,

$$
{}^{r}\mathbf{T}_k = \begin{bmatrix} 0 & -\sin\beta & \cos\beta & x_k \\ -1 & 0 & 0 & y_k \\ 0 & -\cos\beta & -\sin\beta & z_k \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.2}
$$

Here, (x_k, y_k, z_k) is the position of the Kinect with reference to the robot's coordinate frame. β is the angle between Kinect optical axis and horizontal plane. Similarly, we can define the transformation matrix from robot coordinate frame to the ground reference as

$$
{}^{g}\mathbf{T}_r = \begin{bmatrix} \cos\theta_t & -\sin\theta_t & 0 & x_t \\ \sin\theta_t & \cos\theta_t & 0 & y_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.3}
$$

Here, (x_t, y_t, θ_t) is the robot pose in t^{th} time step with respect to the ground reference. Using Equations (5.2) and (5.3), the transformation matrix from Kinect coordinates to ground reference can be obtained,

$$
{}^{g}\mathbf{T}_k = {}^{g}\mathbf{T}_r \times {}^{r}\mathbf{T}_k, \tag{5.4}
$$

$$
{}^{g}\mathbf{T}_k = \begin{bmatrix} \sin\theta_t & -\cos\theta_t\sin\beta & cos\theta_t\cos\beta & x_t + x_k\cos\theta_t - y_k\sin\theta_t \\ -\cos\theta_t & -\sin\theta_t\cos\beta & \sin\theta_t\cos\beta & x_t + y_k\cos\theta_t + x_k\sin\theta_t \\ 0 & -\cos\beta & -\sin\beta & z_k \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.5}
$$

The location vector of the goal with reference to the Kinect coordinate frame is given by,

$$
{}^{k}\mathbf{P} = (x_t^k, y_t^k, z_t^k) \tag{5.6}
$$

Using Equations (5.5) and (5.5), we can obtain the goal location compared to the ground

reference frame,

$$ {}^{g}\mathbf{P} = {}^{g}\mathbf{T}_k \times {}^{k}\mathbf{P} \tag{5.7} $$

$$ \begin{bmatrix} x_t^g \\ y_t^g \\ z_t^g \\ 1 \end{bmatrix} = {}^{g}\mathbf{T}_k \times \begin{bmatrix} x_t^k \\ y_t^k \\ z_t^k \\ 1 \end{bmatrix}, \tag{5.8} $$

$$ \begin{aligned} x_t^g &= x_t^k \sin\theta_t - y_t^k \cos\theta_t \sin\beta + z_t^k \cos\theta_t \cos\beta + x_t + x_k \cos\theta_t - y_k \sin\theta_t, \\ y_t^g &= -x_t^k \cos\theta_t - y_t^k \sin\theta_t \sin\beta + z_t^k \sin\theta_t \cos\beta + y_t + x_k \sin\theta_t + y_k \cos\theta_t. \end{aligned} \tag{5.9} $$

Here, (x_t^g, y_t^g) are the coordinates of the navigation goal in ground reference frame. In this implementation, $x_k = 0$, $y_k = 0.08$ m, $z_k = 1.18$ m, and $\beta = 26.2^o$. Therefore, Equation (5.9) can be further simplified as

$$ \begin{aligned} x_t^g &= x_t^k \sin\theta_t - 0.4421 y_t^k \cos\theta_t + 0.8970 z_t^k \cos\theta_t + x_t - 0.08 \sin\theta_t, \\ y_t^g &= -x_t^k \cos\theta_t - 0.4421 y_t^k \sin\theta_t + 0.8970 z_t^k \sin\theta_t + y_t + 0.08 \cos\theta_t. \end{aligned} \tag{5.10} $$

Using Equation (5.10), the actual location of the navigation goal can be calculated once the robot pose estimation is available. It is combined with current robot pose to decide the navigation target in robot coordinate frame (x_t^r, y_t^r) as illustrated in Figure 4.1.

5.5 Experiment Results

The proposed FCM based SAF for goal detection was tested in an indoor environment using a Kinect sensor. As explained in the introduction of this chapter, the goal of the robot is to detect a cup which is located on a box. In this experiment, red colored boxes and a green colored cup are used for better feature extraction.

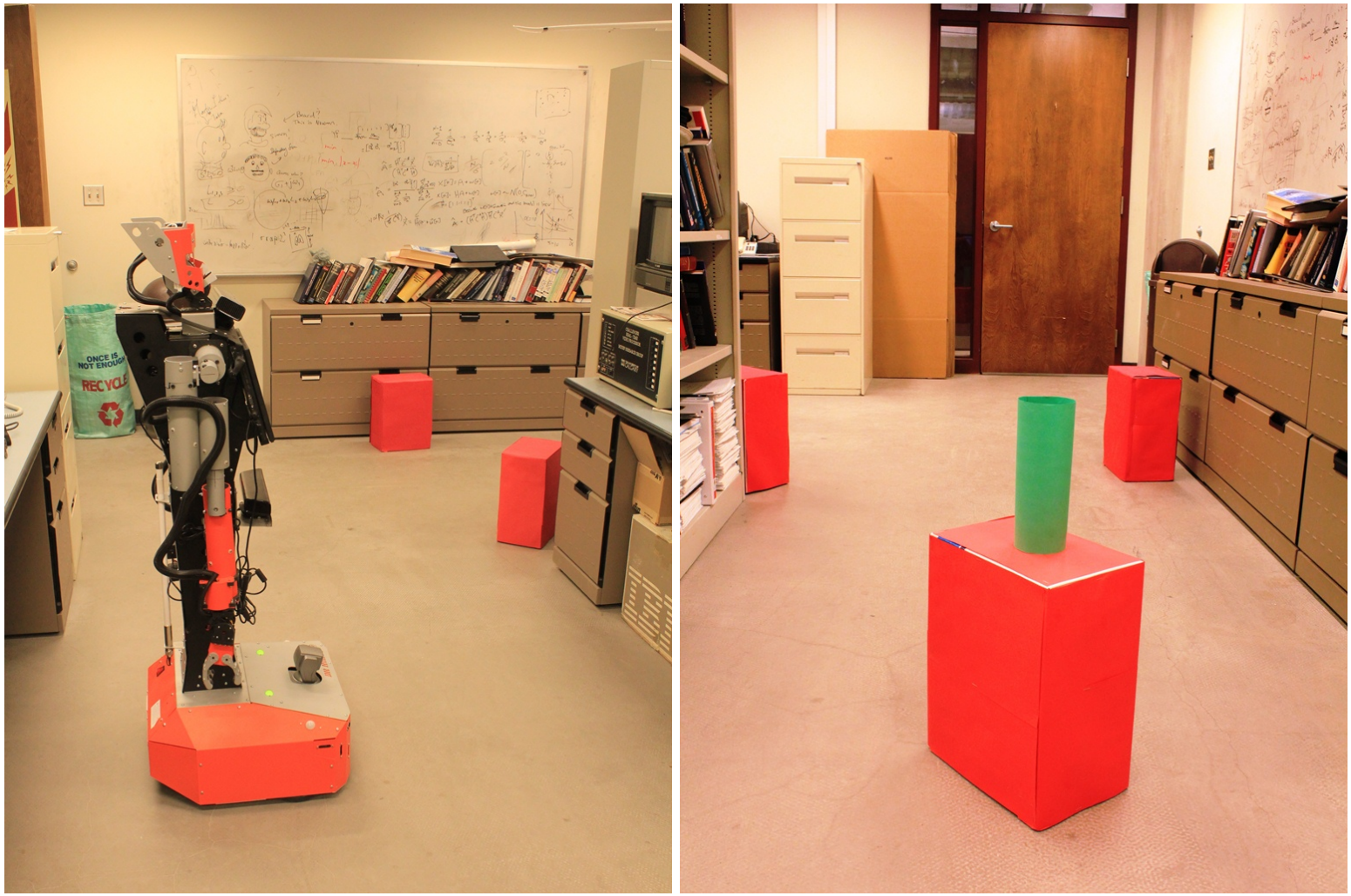
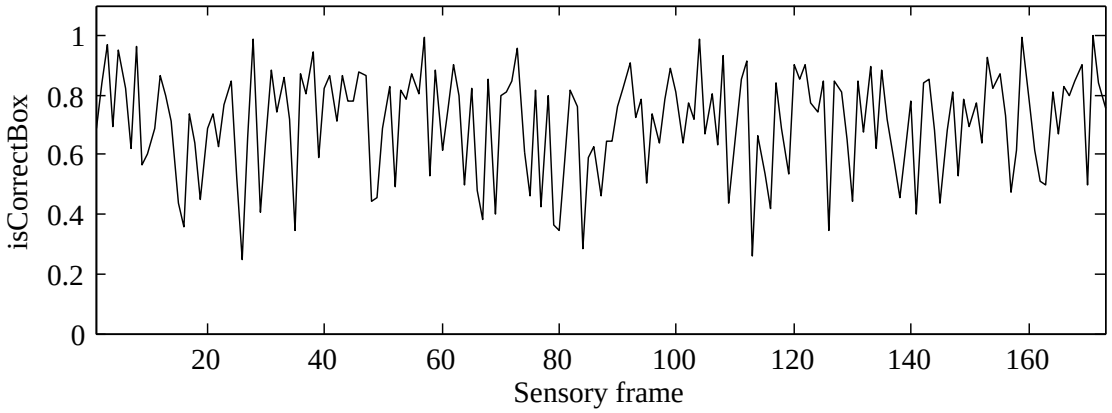
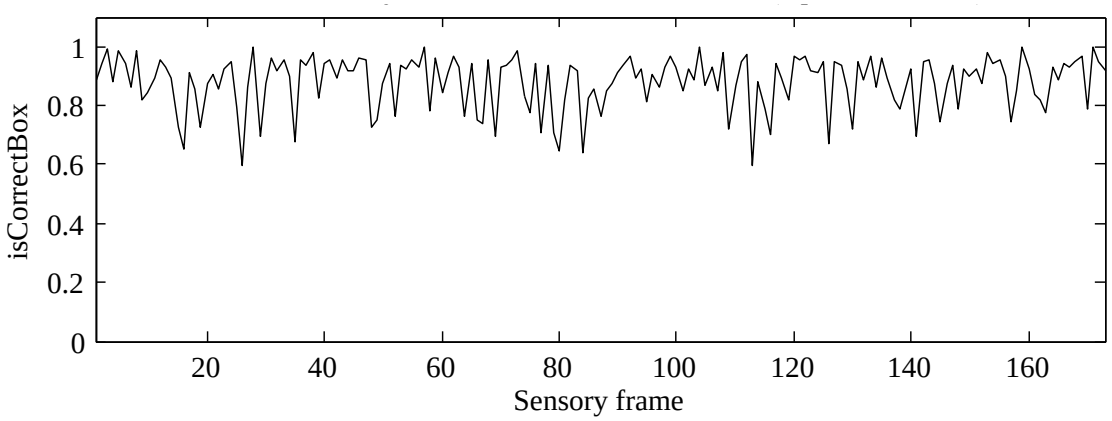


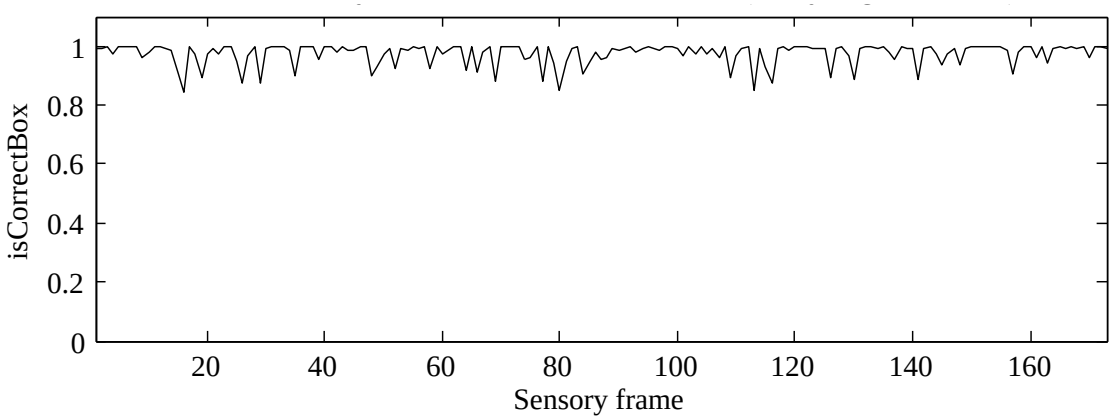
Figure 5.5: Set up of goal detection experiments.



(a)

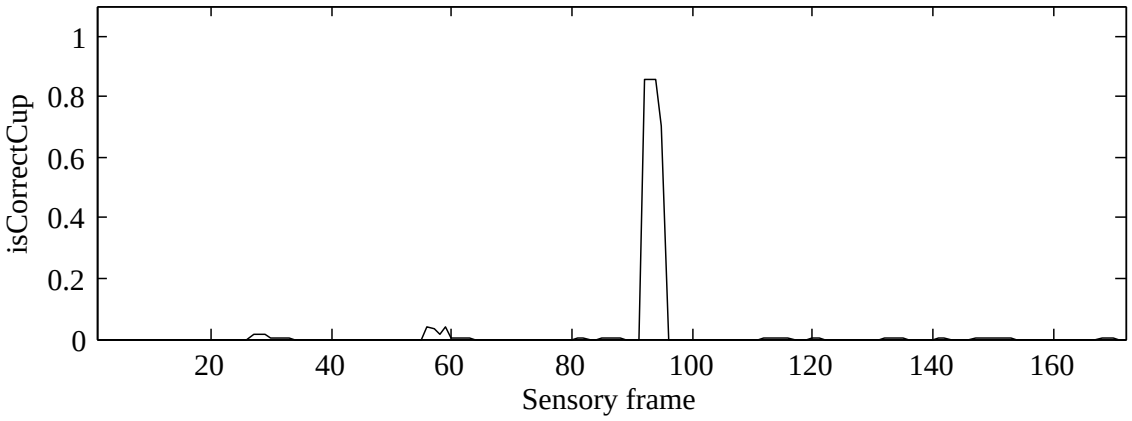


(b)

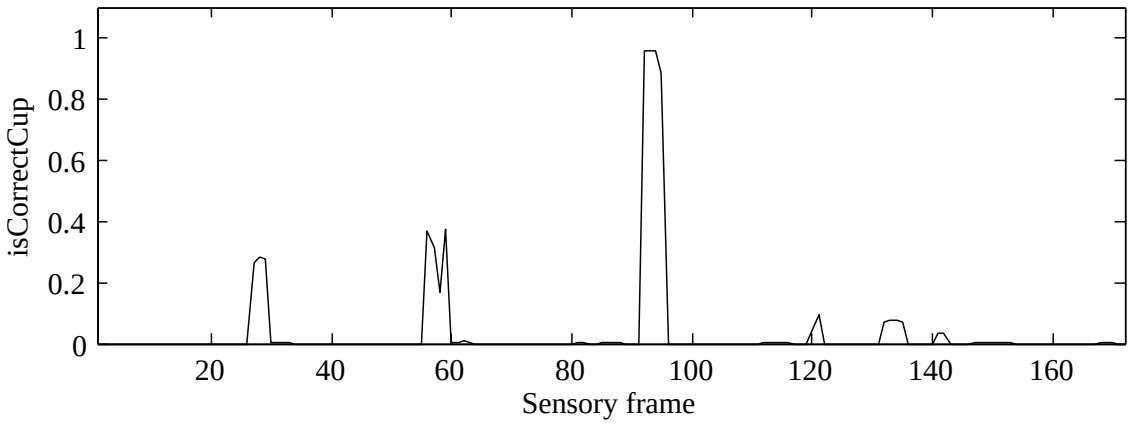


(c)

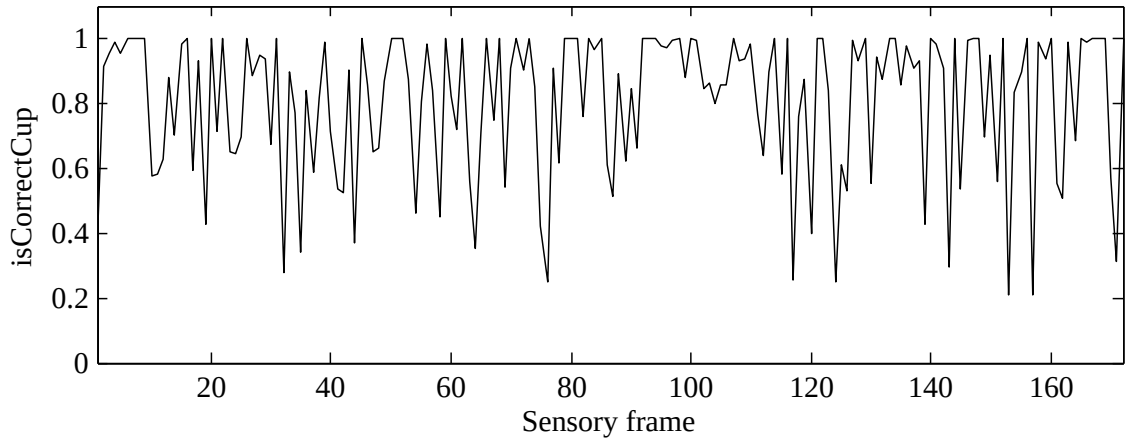
Figure 5.6: Sub-goal assertion confidence for *isCorrectBox* (a) when $\gamma = 0$, (b) when $\gamma = 0.7$, and (c) when $\gamma = 1$.



(a)



(b)



(c)

Figure 5.7: Sub-goal assertion confidence for *isCorrectCup* (a) when $\gamma = 0$, (b) when $\gamma = 0.7$, and (c) when $\gamma = 1$. These results were obtained after removing the cup from the box.

Figure 5.6 illustrates the detection performance of the FCM for the box. The proposed FCM based approach was tested with several γ values. According to the real-time experiment results, the goal assertion confidence of *isCorrectBox* depends on the value selected for γ. In the case of fuzzy algebraic product where $\gamma = 0$, the goal assertion confidence is close to 0, *i.e.* high mismatch, as it needs to correctly match both shape and color of the box. Therefore FCM cannot identify the sub-goal correctly. On the other hand, in the case of fuzzy algebraic sum where $\gamma = 1$, the goal assertion confidence is close to 1 (perfect match) as it only needs to match either shape or color of the box.

However, $\gamma = 1$ can result in false alarms in many practical situations. In order to illustrate that, the cup is moved away from the box and the goal assertion confidence for *isCorrectCup* is calculated. As we can observe from the experimental results in Figure 5.7, goal assertion confidence is close to 1 when $\gamma = 1$, which is simply due to the false alarms. After several experiments, γ is set to 0.7 for this test case. The threshold for goal assertion confidence is selected as 0.75 such that it increases the probability of correct detection and decreases the probability of false alarms. However values of the γ and the threshold depend on the application and need to be tuned carefully.

In the next experiment, the accuracy of the goal location estimation is evaluated. The robot was moved away from the goal 25 cm each time and the distance to the goal was estimated. Figure 5.8 illustrates the statistical results of the experiment. Mean and standard deviation of the estimation error are calculated using minimum of 150 sensory frames at each distance. As illustrated by the results, the estimation error increases with the distance to the goal. Thus the errors can be minimized by estimating the goal location repeatedly while moving towards the goal. However, it should be noted that some parts of the errors are due to measurement errors. Therefore the actual errors are expected to be slightly smaller.

As the last part of the experiment, the overall performance of the goal-driven mobile

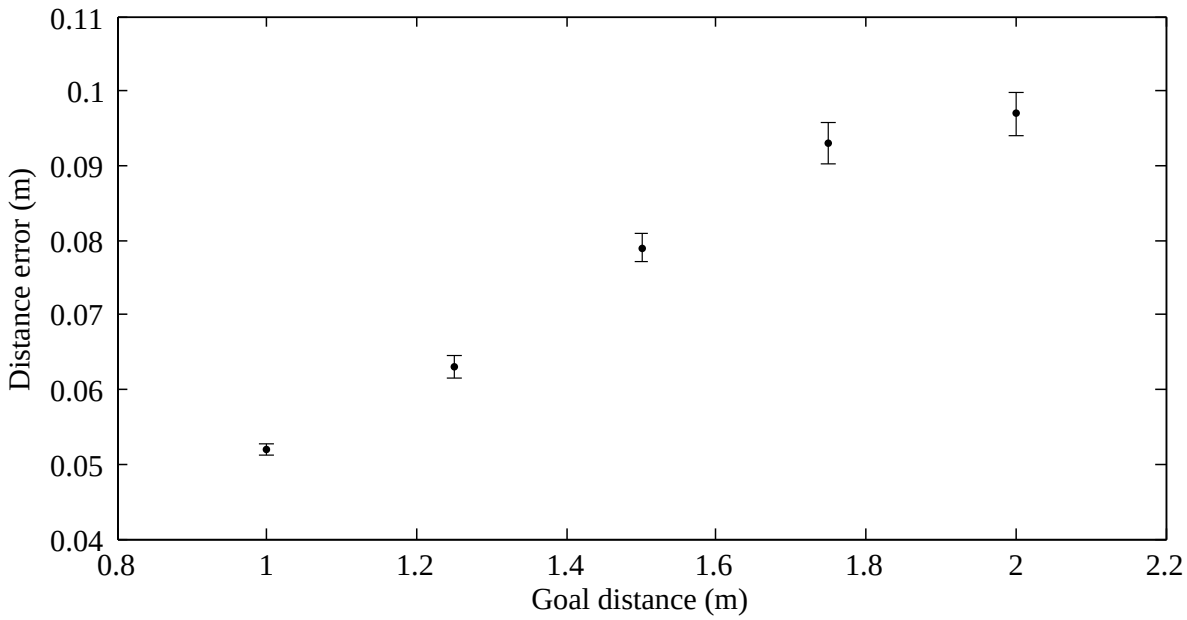


Figure 5.8: Average error of goal location estimation.

robot navigation system is tested. The robot and its navigation goal are arranged such that the robot cannot see the final navigation goal directly from its starting position. The geometrical setup of the experiment is shown in Figure 5.5. The robot has to explore a passage with hard asymmetries and detect its goal. Once it verifies the final navigation goal, the robot obtains the goal position using the homogeneous transformations described in Section 5.4. Then the final navigation goal position is sent to the reactive navigation system as illustrated in Figure 4.1. We set the maximum linear velocity to 0.2 m/s and maximum rotational velocity to 0.34 rad/s (20 deg/s). The robot is asked to stop its navigation when it reaches the final goal within a range of 0.5 m. The occupancy grid map of the test environment and estimated robot trajectory are shown in the Figure 5.9. It also shows the positions of the isolated boxes and a box with a cup (final goal) which are detected during the robot navigation. According to the results obtained, we verify that the proposed high level data fusion system is successfully used for data fusion in navigation goal detection.

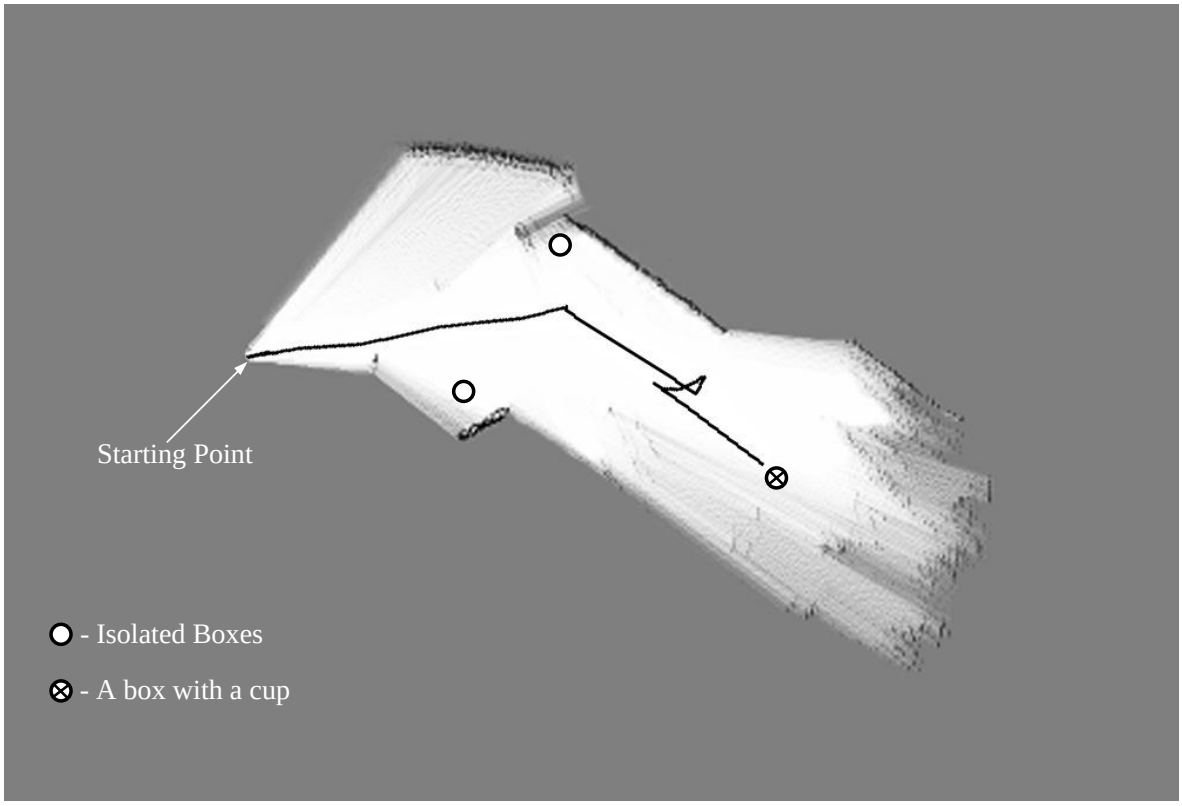


Figure 5.9: Occupancy grid map of the environment with the estimated robot path and goal location.

5.6 Summary

In mobile robotics, navigation goal detection is a challenging problem due to the uncertainty of the environment. Robots need to carefully analyze the situation in order to detect their goals correctly. A fuzzy cognitive map based goal-driven situation assessment framework is presented in this chapter for navigation goal detection. Multiple sensors may be used to observe a single object in the environment and the decision fusion system combines sensory information to verify the sub-goals. Based upon these sub-goals, proposed situation assessment framework operates recursively on the global goal to verify it. The FCM is used as a high level reasoning engine. The goal assertion confidence for each sub-goal is decided using FCM inferences. The navigation goals are verified based on the goal assertion confidence of sub-goals and the rules which connect

sub-goals together. The experimental results presented in Figures 5.6 and 5.7 confirm that the fuzzy gamma fusion can be used effectively with proposed situation assessment framework to maximize the goal detection accuracy while minimizing the number of false alarms. Finally, the experimental results presented in Figures 5.8 and 5.9 have validated the real-time applicability of the proposed goal-driven mobile robot navigation system.

Chapter 6

Conclusion

In most of the real-world applications, dealing with the uncertainties in the environments, sensors, and actuators is challenging for autonomous mobile robots. This thesis proposes three possible solutions to overcome the uncertainties in a goal-driven mobile robot navigation system. Proposed approaches are tested on DrRobot's H20 wheeled mobile robot platform. Odometry and Microsoft Kinect sensors are used as input devices. A detailed discussion on the mobile robot platform and sensors used in the experiments is presented in Chapter 2.

A solution to indoor mobile robot localization problem is presented in Chapter 3. Although global positioning systems (GPS) are popular for mobile robots, it cannot be used in indoor environments. Also the accuracy of non-military GPS is within several meters, which is unacceptable for small scale mobile robots. Odometry is one of the popular solutions for indoor mobile robots. However, odometry measurements accumulate errors due to wheel misalignment, wheel slippage, and uneven wheel diameters. Accuracy of the odometry pose estimation can be improved using auxiliary sensors to observe the environment. Commonly used accurate depth sensors such as laser range finders are, however, expensive to use in most applications.

In the proposed approach, the odometry errors are minimized using a Kinect sensor to observe landmarks in the environment. Kinect is a relatively inexpensive sensor compared to its counterparts. It provides a resolution of 1 cm at a distance of 2 m [63]. Artificial landmarks are distributed in the target environment before experiments. Using artificial landmarks is an inexpensive and feasible solution for indoor environments. The landmark measurements obtained from Kinect are fused with odometry information

using an extended Kalman filter (EKF) and a particle filter (PF). The results of the experiments are presented in Section 3.4. According to the experiment results, pose estimations obtained using the EKF and PF are closer to the ground truth compared to odometry estimations alone. Moreover, PF pose estimations are more robust and accurate compared to EKF pose estimations.

Even though it is possible to use artificial landmarks in some environments, autonomous mobile robots have to frequently operate in unknown environments without any prior knowledge. Also, they have to create maps of the environment on their own and localize themselves in the maps. This has been a challenging problem for mobile robotics community and commonly known as simultaneous localization and mapping (SLAM) problems. However, SLAM does not address the motion control of the robots. Therefore, the necessity of an integrated approach that is capable of localizing, mapping, and motion controlling simultaneously has emerged. This is also known as the mobile robot exploration problem.

In Chapter 4, an integrated mobile robot exploration approach has been proposed. The overall architecture of the proposed system is illustrated in Figure 4.1. In the proposed system, a particle filter with an optimal proposal distribution [86] has been used to solve the SLAM problem. An occupancy grid map is used to represent the robot's knowledge on target environment which does not make any assumption on landmarks. Intermediate navigation goals are selected in the up-to-date map such that the robot can maximize its knowledge on the target environment. The proposed navigation target selection method is discussed in Section 4.3.2. A nearness-diagram (ND) reactive navigation method [87] is used for mobile robot navigation in unknown environments. However, the ND navigation method is only applicable for circular shaped holonomic mobile robots regardless of the fact that most of the mobile robots are non-circular and kinematically constrained. For example, a humanoid wheeled mobile robot was

used in our experiments. Therefore, a trajectory parameter space [88, 89] is used as an abstraction layer of the robot shape and kinematic constraints for the ND method. The proposed integrated exploration system is extensively tested in unknown indoor environments. The experimental results presented in Section 4.5 verify the real-time applicability of the method.

Mobile robots have to determine the location of their navigation goal for motion planning . However, in unknown environments, it is not always possible to locate the navigation goal beforehand. Hence autonomous mobile robots need to detect their targets and localize them using the available sensory information. In real-world applications, navigation goals can be complicated and difficult to identify using a single sensor. Therefore, multiple sensors are utilized with most of the mobile robots. However, due to the ambiguity in mobile robot perception, goal detection has become a challenging problem.

Chapter 5 of this thesis proposes a high level data fusion system for navigation goal detection. A situation assessment framework (SAF) is used to verify the navigation goals based on sensory information and prior knowledge. The proposed architecture of the SAF is shown in Figure 5.1. The navigation goal is first replaced by sub-goals according to the rules stored in the knowledgebase. These sub-goals are verified using fuzzy cognitive map (FCM) inferences. The SAF recursively works on all the sub-goals until it verifies the goal. Depending on the goal assertion confidence of the sub-goals and the rule combining them, a final decision about the goal is made. In this implementation, fuzzy gamma fusion is used as the inference mechanism of the FCMs. Experiment results given in Section 5.5 shows the goal detection capability of the FCM-based SAF.

In summary, three methods are proposed for autonomous mobile robot localization, exploration, and goal detection. All the proposed methods have been implemented on a mobile robot platform and verified through experimental results.

Bibliography

[1] R. Shauri and K. Nonami, "Assembly Manipulation of Small Objects by Dual-arm Manipulator," *Assembly Automation*, vol. 31, no. 3, pp. 263–274, 2011.

[2] P. Kobel and R. Clavel, "Micro Robot for Rotary Desktop Assembly Line," in *Proc. IEEE Int. Symposium on Assembly and Manufacturing (ISAM).* IEEE, 2011, pp. 1–6.

[3] C. Parlitz, W. Baum, U. Reiser, and M. Hägele, "Intuitive Human-machine-interaction and Implementation on a Household Robot Companion," *Human Interface and the Management of Information. Methods, Techniques and Tools in Information Design*, pp. 922–929, 2007.

[4] J. Crowley, "Coordination of Action and Perception in a Surveillance Robot," *IEEE Expert*, vol. 2, no. 4, pp. 32–43, 1987.

[5] A. Aldoma, Z. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. Rusu, S. Gedikli, and M. Vincze, "Tutorial: Point Cloud Library: Three-Dimensional Object Recognition and 6 DOF Pose Estimation," *IEEE Robotics Automation Magazine*, vol. 19, no. 3, pp. 80 –91, Sep. 2012.

[6] S. Naskar, S. Das, A. Seth, and A. Nath, "Application of Radio Frequency Controlled Intelligent Military Robot in Defense," in *Proc. Int. Conf. on Communication Systems and Network Technologies (CSNT).* IEEE, 2011, pp. 396–401.

[7] M. Kim, "Distributed Task Allocation for Multi-robot Systems in Military Domain," Ph.D. dissertation, PURDUE UNIVERSITY, 2011.

[8] L. Gomez, N. Fairfield, A. Szybalski, P. Nemec, and C. Urmson, "Transitioning a Mixed-mode Vehicle to Autonomous Mode," Dec 2011, US Patent 8,078,349.

[9] N. Hockstein, J. Nolan, B. O'Malley Jr, and Y. Woo, "Robotic Microlaryngeal Surgery: A Technical Feasibility Study using the daVinci Surgical Robot and an Airway Mannequin," *The Laryngoscope*, vol. 115, no. 5, pp. 780–785, 2005.

[10] E. Gat, R. Desai, R. Ivlev, J. Loch, and D. Miller, "Behavior Control for Robotic Exploration of Planetary Surfaces," *IEEE Trans. on Robotics and Automation*, vol. 10, no. 4, pp. 490–503, Aug. 1994.

[11] M. Herbert, C. Caillas, E. Krotkov, I. Kweon, and T. Kanade, "Terrain Mapping for a Roving Planetary Explorer," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 2, May 1989, pp. 997–1002.

[12] U. Saranli, M. Buehler, and D. Koditschek, "Rhex: A simple and Highly Mobile Hexapod Robot," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.

[13] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent ASIMO: System Overview and Integration," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 3. IEEE, 2002, pp. 2478–2483.

[14] M. Raibert, K. Blankespoor, G. Nelson, R. Playter *et al.*, "Bigdog, the Rough–terrain Quadruped Robot," in *Proc. of the 17th World Congress*, 2008, pp. 10 823–10 825.

[15] M. Lewis, M. Bunting, B. Salemi, and H. Hoffmann, "Toward Ultra High Speed Locomotors: Design and Test of a Cheetah Robot Hind Limb," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2011, pp. 1990–1996.

[16] L. David, "CURIOSITY ON THE RED PLANET: MARS IS OURS! NASA's Mars Science Laboratory and Its Curiosity Rover's Advanced Instruments Usher

in a New Era in Planetary Exploration," *Aerospace America*, vol. 50, no. 9, p. 26, 2012.

[17] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, 1st ed., ser. Intelligent Robotics and Autonomous Agents. The MIT Press, 2005.

[18] T. Fukuda, Y. Yokoyama, F. Arai, K. Shimojima, S. Ito, Y. Abe, K. Tanaka, and Y. Tanaka, "Navigation System Based on Ceiling Landmark Recognition for Autonomous Mobile Robot-position/Orientation Control by Landmark Recognition with Plus and Minus Primitives," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 2. IEEE, 1996, pp. 1720–1725.

[19] C. Lin and R. Tummala, "Mobile Robot Navigation using Artificial Landmarks," *Journal of Robotic Systems*, vol. 14, no. 2, pp. 93–106, 1997.

[20] F. Giuffrida, C. Massucco, P. Morasso, G. Vercelli, and R. Zaccaria, "Multi-level Navigation using Active Localization System," in *Int. Conf. on Intelligent Robots and Systems, 'Human Robot Interaction and Cooperative Robots'*, vol. 1. IEEE, 1995, pp. 413–418.

[21] E. Kiriy and M. Buehler, "Three-State Extended Kalman Filter for Mobile Robot Localization," McGill University, Montreal, Canada, Tech. Rep., 2002.

[22] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo Localization for Mobile Robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 2, May 1999, pp. 1322 –1328.

[23] J. Leonard and H. Durrant-Whyte, "Mobile Robot Localization by Tracking Geometric Beacons," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 3, pp. 376–382, 1991.

[24] M. Salichs, J. Armingol, L. Moreno, and A. De la Escalera, "Localization System for Mobile Robots in Indoor Environments," *Integrated Computer-Aided Engineering*, vol. 6, no. 4, pp. 303–318, 1999.

[25] R. Kurazume, S. Nagata, and S. Hirose, "Cooperative Positioning with Multiple Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 1994, pp. 1250–1257.

[26] R. Kurazume and S. Hirose, "Study on Cooperative Positioning System: Optimum Moving Strategies for CPS-III," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 4. IEEE, 1998, pp. 2896–2903.

[27] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," in *Proc. of the National Conf. on Artificial Intelligence*. JOHN WILEY & SONS LTD, 1999, pp. 343–349.

[28] D. Schulz and D. Fox, "Bayesian Color Estimation for Adaptive Vision-based Robot Localization," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2. IEEE, 2004, pp. 1884–1889.

[29] S. Lenser and M. Veloso, "Sensor Resetting Localization for Poorly Modelled Mobile Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 2. IEEE, 2000, pp. 1225–1232.

[30] J. Wolf, W. Burgard, and H. Burkhardt, "Robust Vision-based Localization by Combining an Image-retrieval System With Monte Carlo Localization," *IEEE Trans. on Robotics*, vol. 21, no. 2, pp. 208–216, 2005.

[31] B. Kröse, N. Vlassis, and R. Bunschoten, "Omnidirectional Vision for Appearance-based Robot Localization," *Sensor Based Intelligent Robots*, pp. 39–50, 2002.

[32] N. Vlassis, B. Terwijn, and B. Krose, "Auxiliary Particle Filter Robot Localization from High-dimensional Sensor Observations," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1. IEEE, 2002, pp. 7–12.

[33] S. Koenig and R. Simmons, "Exploration with and Without a Map," in *Proc. of the AAAI Workshop on Learning Action Models at the Eleventh National Conference on Artificial Intelligence (AAAI)*, 1993, pp. 28–32.

[34] S. Thrun, "Exploration and Model Building in Mobile Robot Domains," in *Proc. IEEE Int. Conf. on Neural Networks*, 1993, pp. 175–180.

[35] C. Taylor and D. Kriegman, "Exploration Strategies for Mobile Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1993, pp. 248–253.

[36] A. Cassandra, L. Kaelbling, and J. Kurien, "Acting Under Uncertainty: Discrete Bayesian Models for Mobile-Robot Navigation," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, 1996, pp. 963–972.

[37] B. Yamauchi, "A Frontier-based Approach for Autonomous Exploration," in *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation (CIRA)*, 1997, pp. 146–151.

[38] B. Yamauchi, A. Schultz, and W. Adams, "Integrating Exploration and Localization for Mobile Robots," *Adaptive Behavior*, vol. 7, no. 2, pp. 217–229, 1999.

[39] H. Gonzalez-Banos and J. Latombe, "Navigation Strategies for Exploring Indoor Environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[40] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes, "Coordination for Multi-robot Exploration and Mapping," in *Proc.*

the National Conf. on Artificial Intelligence. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2000, pp. 852–858.

[41] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative Multi-robot Exploration," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1, 2000, pp. 476–481.

[42] M. Bernardine Dias, M. Zinck, R. Zlot, and A. Stentz, "Robust Multirobot Coordination in Dynamic Environments," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 4, 2004, pp. 3435–3442.

[43] I. Rekleitis, R. Sim, G. Dudek, and E. Milios, "Collaborative Exploration for Map Construction," in *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, 2001, pp. 296–301.

[44] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated Multi-robot Exploration," *IEEE Trans. on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[45] P. Brass, F. Cabrera-Mora, A. Gasparri, and J. Xiao, "Multirobot Tree and Graph Exploration," *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 707–717, Aug. 2011.

[46] L. Matignon, L. Jeanpierre, and A.-I. Mouaddib, "Distributed Value Functions for Multi-robot Exploration," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2012, pp. 1544–1550.

[47] A. Makarenko, S. Williams, F. Bourgault, and H. Durrant-Whyte, "An Experiment in Integrated Exploration," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 1, Oct. 2002, pp. 534–539.

[48] P. Newman, M. Bosse, and J. Leonard, "Autonomous Feature-based Exploration," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1, 2003, pp.

1234–1240.

[49] R. Sim, G. Dudek, and N. Roy, "Online Control Policy Optimization for Minimizing Map Uncertainty During Exploration," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 2, 2004, pp. 1758–1763.

[50] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte, "Information Based Adaptive Robotic Exploration," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 1, 2002, pp. 540–545.

[51] C. Stachniss, D. Hahnel, and W. Burgard, "Exploration with Active Loop-closing for FastSLAM," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, 2004, pp. 1505–1510.

[52] C. Stachniss, G. Grisetti, and W. Burgard, "Information Gain-based Exploration Using Rao-blackwellized Particle Filters," in *Proc. on Robotics: Science and Systems (RSS)*, 2005, pp. 65–72.

[53] K. Chong and L. Kleeman, "Accurate Odometry and Error Modelling for a Mobile Robot," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, vol. 4, 1997, pp. 2783–2788.

[54] L. Kleeman, "Odometry Error Covariance Estimation for Two Wheel Robot Vehicles," Intelligent Robotics Research Centre, Dept of Electrical and Computer Systems Engineering, Monash University, Tech. Rep. MECSE-95-1, 1995.

[55] C. Pudney, "Surface Following for Manipulators with Proximity Sensors," 1991.

[56] H. Everett, *Sensors for Mobile Robots:Theory and Applications.* Wellesley, MA: A K Peters Ltd., 1995.

[57] R. Siegwart, I. Nourbakhsh, and D. Scaramuzza, *Introducation to Autonomous Mobile Robots*. Cambridge, Massachusetts: MIT Press, 2011.

[58] B. Siciliano and O. Khatib, *Handbook of Robotics*. Springer, 2008.

[59] A. Agent, "The Advantages of Absolute Encoders for Motion Control," *Sensors*, 1991.

[60] P. Bhatti and B. Hannaford, "Single-chip Velocity Measurement System for Incremental Optical Encoders," *IEEE Trans. on Control Systems Technology*, vol. 5, no. 6, pp. 654–661, Nov. 1997.

[61] N. Ganganath and H. Leung, "Mobile Robot Localization Using Odometry and Kinect Sensor," in *Proc. IEEE Int. Conf. on Emerging Signal Processing Applications (ESPA)*, Jan. 2012, pp. 91–94.

[62] P. Rakprayoon, M. Ruchanurucks, and A. Coundoul, "Kinect-based Obstacle Detection for Manipulator," in *IEEE/SICE Int. Symposium on System Integration (SII)*, Dec. 2011, pp. 68–73.

[63] *The PrimeSensor Reference Design*, 1st ed., PrimeSense Ltd., Palo Alto, CA 94301, USA, 2010.

[64] *Scanning Laser Range Finder URG-04LX-UG01 Specifications*, Hokuyo Automatic Co. LTD, Chuo-Ku, Osaka, 540-0028 Japan, Jun. 2009.

[65] *Scanning Laser Range Finder UTM-30LX/LN Specification*, Hokuyo Automatic Co. LTD, Chuo-Ku, Osaka, 540-0028 Japan, May 2009.

[66] *PING Ultrasonic Distance Sensor*, 1st ed., Parallax Inc., Rocklin, California 95765, USA, Jun. 2006.

[67] J.-S. Gutmann and D. Fox, "An Experimental Comparison of Localization Methods Continued," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, vol. 1, 2002, pp. 454–459.

[68] J. Borenstein, H. Everett, L. Feng, and D. Wehe, "Mobile robot positioning-sensors and techniques," DTIC Document, Tech. Rep., 1997.

[69] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*. Cambridge university press, 2010.

[70] R. I, "Cooperative Localization and Multi-robot Exploration," Ph.D. dissertation, School of Computer Science, McGill University, Montreal, Canada, 2003.

[71] P. Maybeck, *Stochastic models, Estimation and Control*. Academic Pr, 1982, vol. 3.

[72] A. Gelb, *Applied Optimal Estimation*. MIT press, 1974.

[73] Openkinect, openkinect main page. Last Accessed on 30-08-2010. [Online]. Available: http://openkinect.org

[74] D. Kerbyson and T. Atherton, "Circle Detection using Hough Transform Filters," in *Proc. Int. Conf. Image Processing and its Applications*, Jul. 1995, pp. 370–374.

[75] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. Wan, "The Unscented Particle Filter," *Advances in Neural Information Processing Systems*, pp. 584–590, 2001.

[76] J. Liu, R. Chen, T. Logvinenko *et al.*, "A theoretical Framework for Sequential Importance Sampling and Resampling," *Sequential Monte Carlo Methods in Practice*, pp. 225–246, 2001.

[77] P. Jensfelt, O. Wijk, D. Austin, and M. Andersson, "Experiments on augmenting condensation for mobile robot localization," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 3. IEEE, 2000, pp. 2518–2524.

[78] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved Particle Filter for Nonlinear Problems," *Proc. IEEE Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 2–7, Feb. 1999.

[79] D. Bapna, E. Rollins, J. Murphy, E. Maimone, W. Whittaker, and D. Wettergreen, "The Atacama Desert Trek: Outcomes," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1, May 1998, pp. 597–604.

[80] S. Smith and S. Dunn, "The Ocean Explorer AUV: A Modular Platform for Coastal Sensor Deployment," in *Proceedings of the Autonomous Vehicles in Mine Countermeasures Symposium*. Naval Postgraguate School, 1995.

[81] L. Whitcomb, "Underwater Robotics: Out of the Research Laboratory and into the Field," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1, 2000, pp. 709–716.

[82] J. Bares and D. Wettergreen, "Dante II: Technical Description, Results, and Lessons Learned," *The International Journal of Robotics Research*, vol. 18, no. 7, pp. 621–649, 1999.

[83] D. Wettergreen, D. Bapna, M. Maimone, and G. Thomas, "Developing Nomad for Robotic Exploration of the Atacama Desert," *Robotics and Autonomous Systems*, vol. 26, no. 2, pp. 127–148, 1999.

[84] S. Thrun, "Robotic Mapping: A Survey," in *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Nebel, Eds. Morgan Kaufmann, 2002.

[85] C. Stachniss, D. Hahnel, and W. Burgard, "Exploration with Active Loop-closing for FastSLAM," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, Sep./Oct. 2004, pp. 1505–1510.

[86] J. Blanco, J. González, and J. Fernández-Madrigal, "Optimal Filtering for Non-parametric Observation Models: Applications to Localization and SLAM," *The International Journal of Robotics Research*, vol. 29, no. 14, pp. 1726–1742, 2010.

[87] J. Minguez and L. Montano, "Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios," *IEEE Trans. on Robotics and Automation*, vol. 20, no. 1, pp. 45–59, Feb. 2004.

[88] J. Minguez, L. Montano, and J. Santos-Victor, "Abstracting Vehicle Shape and Kinematic Constraints from Obstacle Avoidance Methods," *Autonomous Robots*, vol. 20, pp. 43–59, 2006.

[89] J.-L. Blanco, J. Gonzalez, and J.-A. Fernandez-Madrigal, "The Trajectory Parameter Space (TP-Space): A New Space Representation for Non-Holonomic Mobile Robot Reactive Navigation," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2006, pp. 1195–1200.

[90] A. Doucet, S. Godsill, and C. Andrieu, "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering," *Statistics and Computing*, vol. 10, pp. 197–208, 2000.

[91] J. S. Liu, "Metropolized Independent Sampling with Comparisons to Rejection Sampling and Importance Sampling," *Statistics and Computing*, vol. 6, pp. 113–119, 1996.

[92] J.-L. Blanco, J. Gonzalez, and J.-A. Fernandez-Madrigal, "An Optimal Filtering Algorithm for Non-parametric Observation Models in Robot Localization,"

in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2008, pp. 461–466.

[93] T. Lozano-Perez, "A Simple Motion-planning Algorithm for General Robot Manipulators," *IEEE Journal of Robotics and Automation*, vol. 3, no. 3, pp. 224–238, Jun. 1987.

[94] J. Claraco, *Development of Scientific Applications with the Mobile Robot Programming Toolkit (MRPT)*. Machine Perception and Intelligent Robotics Laboratory, University of Málaga, Málaga, Spain, 2010.

[95] M. Endsley, "Toward a Theory of Situation Awareness in Dynamic Systems," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, no. 1, pp. 32–64, 1995.

[96] S. Chandana, H. Leung, E. Bosse, and P. Valin, "Fuzzy cognitive map based situation assessment for coastal surveillance," in *Information Fusion, 2008 11th International Conference on*. IEEE, 2008, pp. 1–6.

[97] M. Hanson, O. Sullivan, and K. Harper, "On-line Situation Assessment for Unmanned Air Vehicles," in *Proc. of the Florida Artificial Intelligence and Research Society (FLAIRS) Conference*, vol. 44, 2001, p. 48.

[98] B. Kosko, "Fuzzy Cognitive Maps," *International Journal of Man-machine Studies*, vol. 24, no. 1, pp. 65–75, 1986.

[99] ——, "Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence," 1992.

[100] M. Styblinski and B. Meyer, "Signal Flow Graphs vs Fuzzy Cognitive Maps in Application to Qualitative Circuit Analysis," *Int. Journal of Man-Machine Studies*,

vol. 35, no. 2, pp. 175–186, 1991.

[101] C. Stylios and P. Groumpos, "Fuzzy Cognitive Maps: A Model for Intelligent Supervisory Control Systems," *Computers in Industry*, vol. 39, no. 3, pp. 229–238, 1999.

[102] Z. Wen-Ran and C. Su-Shing, "A Logical Architecture for Cognitive Maps," in *Proc. IEEE Int. Conf. on Neural Networks*. IEEE, 1988, pp. 231–238.

[103] Z. Li, H. Leung, P. Valin, and H. Wehn, "High Level Data Fusion System for CanCoastWatch," in *Proc. IEEE Int. Conf. on Information Fusion*. IEEE, 2007, pp. 1–6.

[104] A. Steinberg, C. Bowman, and F. White, *Revisions to the JDL Data Fusion Model*. American Inst. of Aeronautics and Astronautics New York, 1998.

[105] J. Llinas, C. Bowman, G. Rogova, A. Steinberg, E. Waltz, and F. White, "Revisiting the JDL Data Fusion Model II," DTIC Document, Tech. Rep., 2004.

[106] C. Stylios, V. Georgopoulos, and P. Groumpos, "The Use of Fuzzy Cognitive Maps in Modeling Systems," in *Proc. IEEE Mediterranean Conference on Control and Systems (MED5)*, vol. 67, 1997.

[107] G. Bonham-Carter, *Geographic Information Systems for Geoscientists: Modelling with GIS*. Pergamon press, 1994, vol. 13.

[108] R. Rusu and S. Cousins, "3D is Here: Point Cloud Library (PCL)," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[109] R. Rusu, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments," Ph.D. dissertation, Computer Science Dept., Thechnical University of Munich, Germany, Oct. 2009.

[110] R. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2010, pp. 2155–2162.

[111] M. Muja and D. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," in *Proc. Int. Conf. on Computer Vision Theory and Applications*, 2009, pp. 331–340.