Incorporating Sensor Uncertainty in Robot Map Building using Fuzzy Boundary Representation

by

Alejandro Tovar

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfilment of the requirements of the degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

Faculty of Engineering

University of Manitoba

Winnipeg

Abstract

A map is important for autonomous mobile robots to traverse an environment safely and efficiently through highly competent abilities in path planning, navigation and localization. Maps are generated from sensors data. However, sensor uncertainties affect the mapping process and thus influence the performance of path planning, navigation and localization capabilities. This thesis proposes to incorporate sensor uncertainty information in robot environmental map using Fuzzy Boundary Representation (B-rep). Fuzzy B-rep map is generated by first converting measured range data into scan polygons, then combining scan polygons into resultant robot B-rep map by union operation and finally fuzzifying the B-rep map by sweeping sensor uncertainty membership function along generated B-rep map. A map of the fifth floor of E1 building is generated using the proposed method to demonstrate the alleviation in computational and memory load for robot environment mapping using Fuzzy B-rep, in contrast to the conventional grid based mapping methods.

Acknowledgments

First of all, I would like to express my special gratitude to Dr. Wai-Keung Fung for his support, guidance and for sharing his experience for the successful completion of this research work. It was a privilege to work under his supervision.

Thanks to the administrative staff of the University of Manitoba, for their support in each step of this study, especially to Amy Dario.

I would also like to thank my colleagues. I am especially grateful to Mr. Francisco Gomez for assisting me in many different ways through all the graduate studies.

I wish to thank my parents Gessy Nunez de Tovar and Jesus Tovar. Without them, I would not be the person I am today.

Finally, I wish to thank my brother and sister, especially my sister Gessyca Tovar for her encouragement and suggestions in writing this thesis.

Dedication

To my lovely parents

Contents

Front Matter

List of Tables

List of Figures

Chapter 1

Introduction

Safe navigation and reliable localization are two elementary capabilities necessary for autonomous mobile robots to traverse in an unknown environment. Robot localization mainly depends on the map generated manually or based on sensor data automatically. Localization is achieved by matching the map information with recent robot perception of the environment. Common map building approaches are topological maps [1] and occupancy grid [2]. Topological mapping maintains robot environment information using graphs. Properties of landmarks or features in robot environment are stored in graph nodes. The spatial relationship among landmarks or features is represented using graph links. The main problem of this method is that it has difficulties in representing open area in robot environments. Geometric information of the represented robot environments is usually missing in topological maps. Therefore, the robot does not have geometric information about the occupancy of an area. Topological maps usually do not consider sensor uncertainty caused by measurements errors or interference from the environment. The sensor uncertainty affects the accuracy of the generated maps in representation of robot environment and spatial reasoning. On the other hand, occupancy grid based methods represents robot environments as an evenly spaced field (called cells) of random variables in special cells. Each cell contains the probability of being occupied in the corresponding robot environment region [2] [3]. An occupancy grid map only contains geometric information of the modeled environment. The disadvantage of an occupancy grid map is that it does not contain topological information and robot path planning becomes difficult, as sophisticated algorithms are needed to extract topological information of the environments from geometric maps.

This research is expected to have high impact in the robotics community by developing a unified map representation that is capable of handling sensor uncertainty for map building, path planning, navigation, and localization for mobile robots. This thesis focuses on 2D map building for mobile robots. This thesis proposes to represent robot environment maps using boundary representation (B-rep) and fuzzy set theory to incorporate the robot sensor uncertainty into the maps. A fuzzy boundary representation map is a combination of a B-rep map (a collection of boundary information of open area of robot environment) with fuzzy set theory for modeling the sensors uncertainties. The idea of fuzzy map building has already been introduced to the robotics community [4]. However, there is still much space for improvement, like generating a more efficient fuzzy algorithm or fuzzifying a different map type like boundary representation. This thesis suggests that robots do not have to implement sophisticated algorithms to enhance accuracy in mapping, navigation, path planning, and localization. Instead, this thesis proposes to include the uncertainty information due to imperfect sensors as safety tolerance in robot maps. Thus, the robots can produce task plans and execute commanded tasks appropriately and securely considering the safety margin stored in the fuzzy boundary of robot maps for path planning, navigation and localization.

1.1. Motivation and Challenges

Autonomous robots require environment information to traverse and complete tasks. When the robot is situated in a novel environment, the first thing it does is to collect information of the environment using onboard sensors. By using raw data captured by sensors, robots can create environment maps for path planning, navigation and localization. A map is required for path planning because it provides information of navigable space and obstacles in the environment. For localization, the map information is matched with current perception of the environment to obtain the position and orientation of the robot. Thus, map building is an important competence in autonomous robots. However, sensor data uncertainty introduces inaccurate information in robot map building. This uncertainty in the data is caused by many reasons, and it is highly dependent on the sensor used [5].

Probabilistic methods have been widely studied for reduction of the effect of sensor measurement error to robotic applications, including map building [3] [5]. These methods work on the assumption that the sensor measurement error follows a Gaussian distribution and use the Gaussian model to compensate the effect of this error, such as the Kalman Filter [6] [7]. Studies on probabilistic methods have introduced nonlinear methods, like Extended Kalman Filter and the Unscented Kalman filter [8] to handle nonlinear sensor measurement errors. These methods can work on non-Gaussian distribution of measurement errors and nonlinear processes, at the cost of high computational complexity $O(n^3)$ [9]. Real-time implementation of these techniques is difficult on resource (in terms of computational power, memory and power) limited hardware platforms. On the other hand, this thesis investigates the possibility of incorporating the sensor uncertainty information in robot maps. Instead of finding a way to compensate the effect of the measurement errors on the path planning, navigation and localization abilities of autonomous robots, this research proposes to incorporate the uncertainty information directly into the map. Various ways to represent uncertainties are briefly described in Section 2.3. Specifically, the sensor impreciseness is incorporated in robot maps using the fuzzy set theory, which allows representation of the vague and imprecise entities. This thesis focuses on generation of fuzzy boundary representation of robot environments.

In robot environmental map building, the first difficulty encountered is to create the crisp boundary representation map based on range sensor data. When the size of the modeled environment is large, the robot or multiple robots need to take several measurements of the environment that leads to the challenge of efficient combination of different scans of sensor information for the building of a global robot map. Another challenge is how the generated crisp boundary representation map of a robot environment is transformed into its fuzzy counterpart so as to reflect sensor uncertainty inherited in robot sensors.

1.2. Proposed Approach

This thesis proposes to represent robot environments using fuzzy boundary representation (fuzzy B-rep) based on uncertain range data and odometry data. The first step is to convert raw sensor data into a crisp boundary representation map (B-rep map), where the robot environment is represented by the boundary of open area of the environment. Boundaries are represented by a series of lines extracted by the Hough Transform from raw range sensor data. Lines found by the Hough Transform algorithm are then organized to form the edges of a scan polygon. Multiple scan polygons generated at different locations can be combined to form the resultant environment map by performing the union operation of them. Obstacles in the environment are represented by holes in the principal polygon that represents the boundary of the robot environment. Detail of hole detection and management is discussed in Chapter 3.

A fuzzy boundary representation map of the robot environment is generated by sweeping sensor uncertainty fuzzy set along the boundary of its crisp boundary representation map generated from sensor data. The sensor uncertainty fuzzy set represent the impreciseness of the range data and odometry data. Details of fuzzy boundary representation (fuzzy B-rep) generation are described in sections 4.1 and 4.2.

1.3. Contributions

The advantages of modeling the environment map using fuzzy boundary representation are that the robot has access to both, topological information and geometric information of its environment. It leads to safe navigation and path planning because it takes into consideration the uncertainty of sensor measurements.

The following lists the contributions of this thesis:

- A way to incorporate sensor data impreciseness inside the maps is developed.
- An algorithm for sweeping the sensor uncertainty membership function along the environment boundary representation for accelerated α-cuts calculations is developed.
- A fuzzy B-rep map data structure for 2D robot maps is developed.
- The research provides an efficient approximation for the fuzzification of the environment of the open area of robot environment.

1.4. Thesis Organization

The thesis is organized as follows. The background information is presented in Chapter 2, where information about fuzzy set theory, different types of map representations, and a variety of ways to handle uncertainties are presented. Chapter 3 explains how to create the crisp boundary representation map (B-rep map), including how to convert raw sensor data to B-rep data structure, and how to handle special features of the B-rep maps like holes (as obstacles in robot environments). In Chapter 4, conversion of crisp B-rep maps to fuzzy B-rep maps is investigated. Selection of the sensor uncertainty membership functions, algorithms of sweeping operation and the data structure for fuzzy B-rep maps are discussed in this chapter. In Chapter 5, experimental results and discussions are presented, and finally chapter 6 concludes the thesis and suggests future works.

Chapter 2

Background

This chapter presents the background of the research of building robot environment map using fuzzy boundary representation. We present the basics of fuzzy set theory, discuss different methods of robot environmental mapping, and briefly describe different ways to represent uncertainties.

2.1. Basics of fuzzy sets

Fuzzy sets are sets with boundaries that are not precise, this mean that the membership of elements in a fuzzy set is not a matter of affirmation or denial (True or False), but a matter of degree (Lofti A. Zadeh [10]). In contrast, a crisp set is defined in a way as to dichotomize the individuals in some given universe of discourse into two groups: members (those belong to the set) and nonmembers (those that do not). There is a clear distinction between the members and nonmembers of the set. However, in a fuzzy set, instead of taking such sharp set boundaries in the universe of discourse, assigns to each possible element a grade of membership. George and Klir [11] stated that:

> "The capability of fuzzy sets to express gradual transitions from membership to nonmembership and vice versa has a broad utility. It provides us not only with a meaningful and powerful representation of measurement uncertainties, but also with a meaningful representation of vague concepts expressed in natural language."

Let us take the YOUNGNESS of a person as an example. Given a person x, in the set of all people on earth (the universe of discourse under this context), age(x) gives his/her age. The membership function of the fuzzy set YOUNG, defined in the universe of discourse X is expressed as:

$$\left\{\mu_{young}(x) = 1\ if\ age(x) \leq 20; \frac{30-age(x)}{10} if\ age(x) \leq 30; 0\ if\ age(x) > 30\right\}$$

As shown in Figure 2.1. If a person is 25 years old, he or she does not belong entirely in the YOUNG fuzzy set but partially with a membership grade of 0.5.

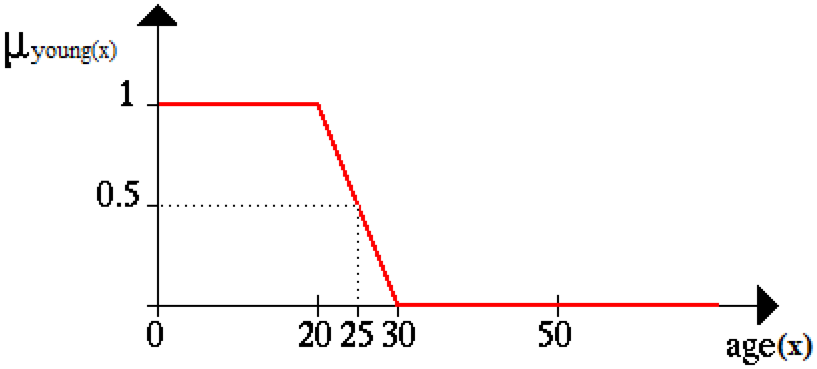

Figure 2.1 Fuzzy membership function for YOUNGNESS

2.1.1 Membership function

A membership function is a characteristic function of a fuzzy set. It shows the membership of an element in the subset A of the universe of discourse X and is denoted by $\mu_A(x)$. For example, Figure 2.2 illustrates the membership functions of four heat conditions in a room namely COLD, COOL, WARM, and HOT. Each fuzzy subset has its own membership function, and some of them overlap. That means that in this universe of discourse, a temperature value may have different grades of memberships in two different subsets. Consider the case when the room temperature is 0°C, we observe that this value has a membership of 0.5 in the COLD subset and a membership of 0.5 in the COOL subset in Figure 2.2. The membership grade of each element in a fuzzy set has to be in the range [0, 1].

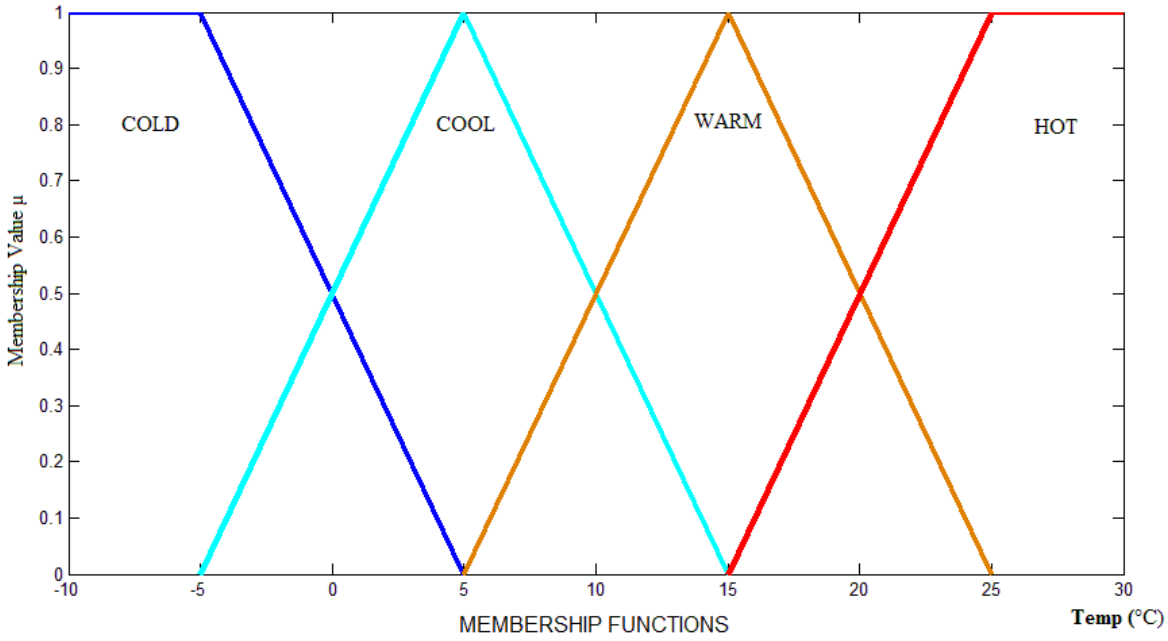

Figure 2.2 Fuzzy sets to characterize the Temperature of a Room

Figure 2.3 shows the possible descriptions of fuzzy memberships for characterizing the concept of “a real number that is close to 2.” However, it is worth to mention that the membership function shape depends on the characteristics of the uncertainty in the process and the nature of the applications. Each function in Figure 2.3 is a member of a family of parameterized functions. The followings are general formulas describing the four families of membership functions, where r denotes the core of the fuzzy set with the membership grade equals 1 (i.e., r=2 for all functions in Figure 2.3), and p_i $(i \in N)$ is the parameter that determines the rate at which, for each x, the function decreases with the increasing difference $|r-x|$ [11] [12]:

$$A_1(x)=\begin{cases} p_1(x-r)+1 & when\ x\ \in \left[\frac{r-1}{p_1},r\right) \\ p_1(r-x)+1 & when\ x\ \in \left[r,\frac{r+1}{p_1}\right] \\ 0 & otherwise \end{cases}$$

$$A_2(x)=\left\{\frac{1}{1+\ p_2(x-r)^2}\right.$$

$$A_3(x)=\left\{e^{-|p_3(x-r)|}\right.$$

$$A_4(x)=\begin{cases} \frac{1+\cos\left(p_4\pi(x-r)\right)}{2} & when\ x\ \in \left[\frac{r-1}{p_4},\frac{r+1}{p_4}\right] \\ & 0\ otherwise \end{cases}$$

For each $i=1,2,3,4$, p_i increases while the graph A_i becomes narrower. Functions in Figure 2.3 exemplify these classes of functions for $p_1=1, p_2=10, p_3=5, p_4=2, and\ r=2$. [11] [12].

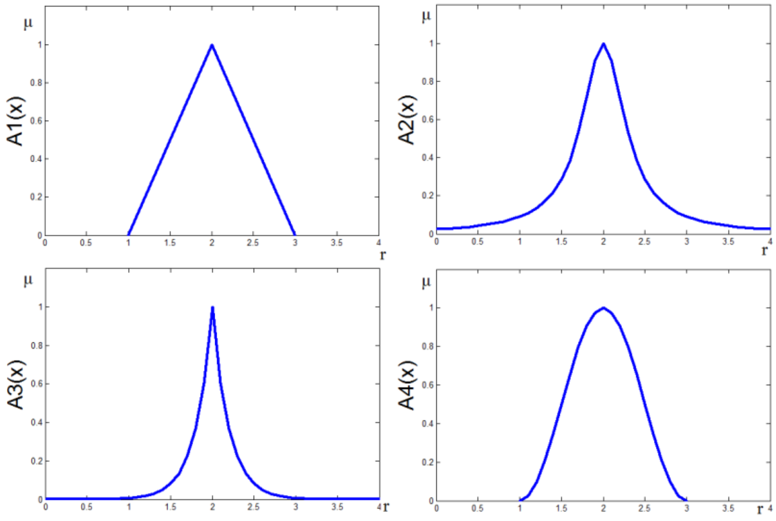

Figure 2.3 Examples of membership functions for real number close to 2

2.1.2 α-cuts

The α-cut of a fuzzy set A is a crisp set that contains all the elements whose degree of membership in A is no less than α [11] [12]. A α-cut is a level set of the fuzzy set membership function and is defined as:

$$A_\alpha = \{x \in X | \mu_A(x) \geq \alpha\}.$$

Similar to the α-cut, the strong α-cut (A_α^+) is a crisp set that contains all the elements whose degree of membership in a fuzzy set A is strictly greater than α and is,

$$A_\alpha^+ = \{x \in X | \mu_A(x) > \alpha\}.$$

The *support* of a fuzzy set A in a universe of discourse X is the crisp set that includes all the elements whose degree of membership is greater than zero. In other words, the *support* is the strong α-cut when α=0. Formally,

$$supp(A) = \{x \in X | \mu_A(x) > 0\}$$

In contrast, the *core* of a fuzzy set A is represented by the α-cut when α=1. Formally,

$$core(A) = \{x \in X | \mu_A(x) = 1\}$$

For example, for the membership function $A_1(X)$ shown in Figure 2.3, the *support*, *core*, and the α-cuts $A_{0.3}$ *and* $A_{0.7}$ are shown in Figure 2.4.

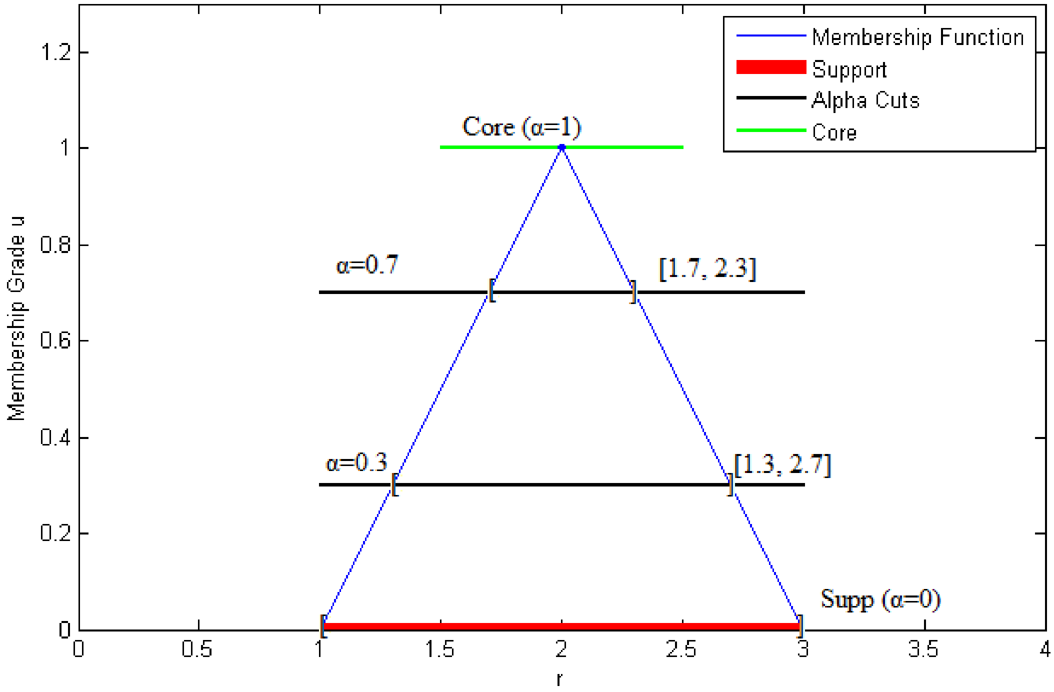

Figure 2.4 Support, Core and α-cuts (α=0.3, α=0.7) of membership function A1(X) of Figure 2.3

2.1.3 Fuzzy set operations

In this section, several set operations for fuzzy sets are defined. Let A and B be two fuzzy sets in the universe of discourse U.

1. **Complement** denoted as $\bar{A}$, is defined as: $\mu_{\bar{A}}(x) = 1 - \mu_A(x)\ \forall\ x\ \in U$
2. **Intersection:** $\mu_{A \cap B}(x) = \min[\mu_A(x), \mu_A(x)]\ \ \forall\ x\ \in U$
3. **Union:** $\mu_{A \cup B}(x) = \max[\mu_A(x), \mu_A(x)]\ \ \forall\ x\ \in U$

Many other fuzzy sets operations exist. However, they are not relevant to the objectives of this thesis (see [13] for details).

2.1.4 The Extension Principle

The extension principle is a fuzzy set theory that is used for fuzzifying crisp functions. Its purpose is to generalize crisp mathematical mappings in fuzzy sets. Suppose that A is a fuzzy set on the universe of discourse X, B is a fuzzy set in the universe of discourse Y, and f is a function for mapping X to Y, $f: X \rightarrow Y$.

A is defined as

$$A = \{\mu_A(x_1)/(x_1), \mu_A(x_2)/(x_2), \dots, \mu_A(x_n)/(x_n)\} \; x_i \in X$$

The extension principle states that the image of fuzzy set A under the mapping $f(A)$ can be expressed as a fuzzy set B.

$$B = f(A) = \{\mu_A(x_1)/(y_1), \mu_A(x_2)/(y_2), \dots, \mu_A(x_n)/(y_n)\}$$

Where $y_i = f(x_i)$, $y_i \in Y \;\; i = i = 1, 2, 3, \dots, n$.

For any x_h *and* x_j, where $h \neq j$, and their images $f(x_h)$ *and* $f(x_j)$ are equal so that $f(x_h) = f(x_j) = \eta$, the membership grade of $\mu_B(\eta)$ is defined as $\mu_B(\eta) = \sup(\mu_A(x_h), \mu_A(x_j))$.

The membership function of the fuzzy set B is defined as,

$$\mu_B = \begin{cases} \sup \mu_A(x) & for\, y = f(x), \\ 0 & for\, y \neq f(x). \end{cases}$$

Fuzzification of functions or mapping using the Extension Principle including fuzzifying crisp boundary representation of robot environment maps usually involves heavy computational processes. In order to reduce the computational complexity in fuzzifying boundary representation, this thesis proposes an efficient approximation to the boundary fuzzification process in fuzzy B-rep map generation.

2.2. Robot Map Representations

Robot environment map representation can be classified by three categories, raw sensor patterns; landmarks-based maps, navigability of space maps [14]. Some of their strengths and limitations are briefly explained.

2.2.1. Raw sensor patterns: Sensor data patterns captured at different discrete locations are stored in robots. A new location is found when the aggregated difference, with respect to various environment specific metrics, between the current view and the views obtained at previous location is higher than a predefined threshold [15]. For instances, the works of Cheng [16] and Hermer [17] both proposed that some animals memorize specific views for navigation. Franks *et al.* devised this biologically inspired map type for robot navigation [18]. Nevertheless, the use of raw sensor patterns has problems with the sensors uncertainties and recognizing one sensor view from another, so rather than to be used for map representation they are more useful to identify features in a global map.

2.2.2. Landmarks: Landmarks are basic objects or features in the environment that are easy to identify. They can be represented by their positions or signatures. Landmarks map type simplifies human style navigation. Denis [19] focused on the description of routes. He concluded that humans employ a rich repertoire of landmarks, e.g., “the hardware store,” “the lighthouse,” etc. The environmental map grows with the number of landmarks in the environment. Localization using these maps is easy because the robot will only have to identify the closest landmark to know its position. A common problem in this type of mapping information is how to identify a landmark when it is captured from various perspectives. Landmark identifications are especially difficult with image sensors, because objects may look completely different or occluded from different perspectives. Nevertheless, new algorithms in image processing described in [20] allows for efficient topology estimation in optical mapping. One solution for localization is to keep track of the previous landmark with the odometry data, and therefore the robot has to have its position updated at any moment. This leads to recent mobile robotics research problem called Simultaneous Localization and Mapping (SLAM) [3]. A SLAM problem is usually solved using Kalman Filter [6] [15] [21] or the particle filter [15]. However, the need for the robot to keep track of previous landmarks adds extra computational and memory loads while the quantity of landmarks increases and thus renders this solution inappropriate for large maps in robots built with limited resources.

2.2.3. Navigability of space: This type of map specifies what part of the environment is navigable for robots. It can be further classified in three categories, occupancy grids map, topological map, and boundary of navigable space.

A. Occupancy Grids

An Occupancy grid map represents the robot environment as an evenly spaced field of random variables. Each cell holds the degree of occupancy of obstacles inside it. The simplest form of an occupancy grid is the binary occupancy grid so that each cell has value of one for empty space and zero for occupied space or vice versa.

Moreover, current occupancy grid maps employ probabilistic algorithms to estimate the probability of occupancy of each indexed grid cell [3]. Specially, Elfes [22] used Bayesian estimation procedures to update the probability of occupancy of each cell from readings taken by different sensors and perspectives.

In spite of the fact that an occupancy grid map is one of the most commonly used mapping methods in mobile robotics; it still has some limitations. One limitation is that the representations do not contain any topological information of the environment (it only contains the degree of occupancy of an array of cells), and another is that robot path planning can be difficult without implementation of complex algorithms. Memory requirements depend on the physical dimension of the environment and the resolution of the cell size in the map, and this is a drawback when mapping large environments in robots with limited hardware (Table 5.3). For example, an occupancy grids map with cell size of 50cm^2 for a 100m^2 environment is depicted in Figure 2.5. This map contains the occupancy probability of 400 cells. Black cells denote high probability of occupancy, gray cells denote medium probability of occupancy, and white cells represent free space and red cell is the current robot position.

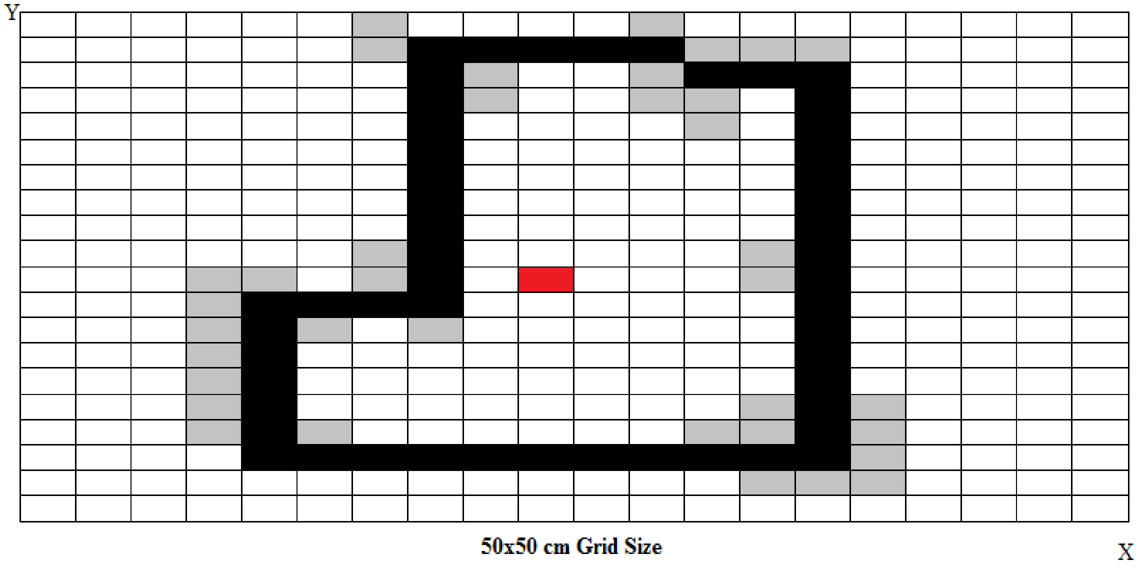

Figure 2.5 400 cells in typical Occupancy grid map

B. Topological map

It is also called roadmap. It represents routes in the environment using a variant of Generalized Voronoi Diagram (GVD) [23]. The GVD represents the medial axis of free space (“skeleton”). For all points x in the Euclidean space, there is one point S closest to x, where S represents a d-dimensional sphere inscribed in free space [23]. Each point in the GVD is the center of a sphere that touches at least two points of the obstacle outline. A graph called Generalized Voronoi Graph (GVG) is then derived from GVD, which has information of meet points and end points of the GVD in the nodes of GVG. For example, Figure 2.6 shows a GVD representation of the blueprint of the fifth floor of the E1 building of the University of Manitoba and its corresponding GVG. The GVD shows the “skeleton” of the blueprint and the GVG represents GVD in a series of vertex points or nodes linked together as a graph.

GVG offers compact representation of robot environment. Furthermore, routes that follow the GVD are maximally safe as they maintain maximum distance to obstacles and are especially popular for indoor office scenarios [23]. However, GVG is highly susceptible to sensor uncertainties. Small changes in the environment can alter the graph basic structure, and sophisticated algorithms for recognition of environment change are required for robust navigation of autonomous robots in the environment.

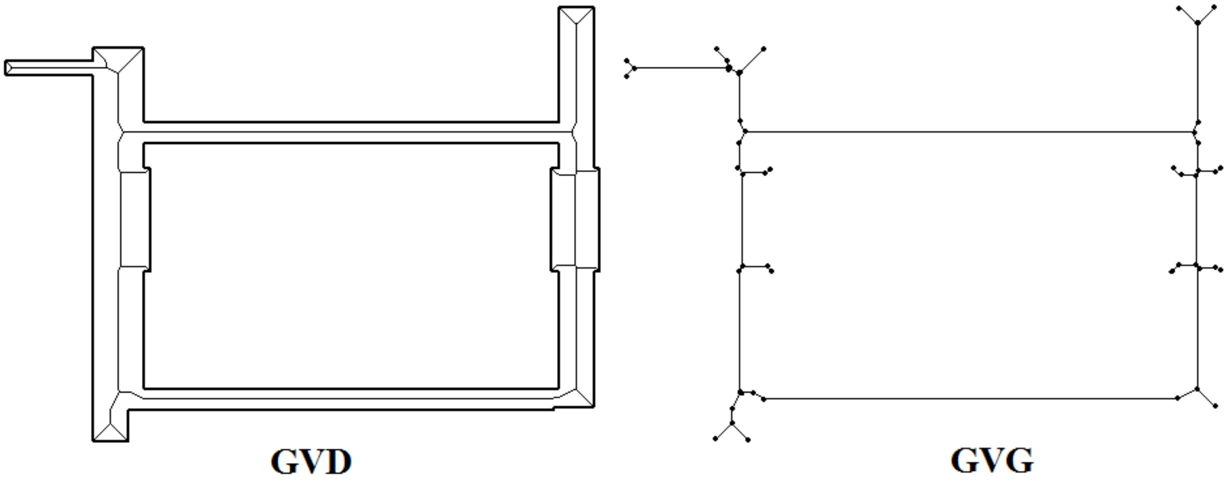

Figure 2.6 GVD and the corresponding GVG of the fifth floor of the E1 building blueprint

C. Boundary of navigable space (Boundary representation or B-rep)

This type of map represents the environment by the boundaries of the free space. It only represents boundaries of obstacles and walls in the environment. Information taken from the laser range finder can be extracted to identify boundaries of navigable space.

This mapping technique is particularly useful for indoor environments. The main merits for using B-rep map style for representing the environment is that it contains both topological and geometrical information of the robot environment, which leads to easy application of common path planning algorithms. Moreover, the memory requirement is small in comparison to occupancy grid map style and is independent of the size of the environment (See Table 5.3). Thrun [24] [25], mentioned additional advantages of B-rep maps over occupancy grid maps:

- B-rep maps can be more accurate, since the grid sizes in occupancy grid maps are trade-offs between accuracy, compactness, and efficiency.
- B-rep maps are more robust in handling dynamic environments, because moving objects in the real world corresponds to holes in the B-rep map. Therefore, predicting the movement direction of the object is possible. On the other hand, occupancy grid maps do not have the knowledge of real world objects, which complicates the handling of dynamics of the environment in the map.

This type of map is selected in this thesis. Further information about boundary representation is given on Chapter 3.

2.3. Representing Uncertainty

Before describing details of different ways to represent uncertainties, is it essential to define what kind of uncertainty is handled in the research. Section 4.1 refers to the proper selection of the sensor uncertainty membership function; in this section the odometry sensor and the laser range finder sensor measurement errors are the principal sources of uncertainty. The uncertainty to be represented is the one produced by the sensors measurement errors.

2.3.1. Probabilistic approaches

This is the most commonly used method for handling uncertainties in robot mapping and localization. Probabilistic algorithms approach the problem by explicitly modeling different sources of sensor uncertainties and their effects on the measurements. These approaches provide quantitative measures of error obtained. Most probabilistic methods work well with Gaussian distribution of the sensor uncertainties [6]. However, real world uncertainty is not always Gaussian. Navigation and localization algorithms, based on the assumption that the sensor errors follow a Gaussian distribution, often fail under the non – Gaussian uncertainty environment in real world. Probabilistic methods, such as the Extended Kalman filter and the Unscented Kalman filter [8] can work with non-Gaussian processes and they use nonlinear functions to model sensor uncertainty to improve the accuracy of the result. However, using non-linear models for pose estimation, increases computational complexity $O(n^3)$ (when using Unscented Kalman Filter [9]). Another limitation of probabilistic approaches is the difficulty in fusing data from multiple sensor modalities. Probabilistic data fusion requires understanding of the sensed environment, the type of measures, the characteristics of the sensors, and finally the kind of uncertainty introduced by the sensors [26]. For an instance, laser range finder (LRF) and sonar sensors are two common types of sensors installed on mobile robots. It is well known that the accuracy of LRF is greater than that of the sonar sensor. However, in environments where the laser reflectivity is low, the accuracy of the LRF is impaired and the sonar range readings have higher accuracy.

2.3.2. Rough Sets

The theory of rough sets is motivated by the practical needs to interpret, characterize, represent, and the process indiscernibility of individuals. Rough set theory provides a systematic method for representing and processing vague concepts caused by indiscernibility in situations with incomplete information or the lack of knowledge [27]. For example, a group of students registers for a collection of courses, many students would belong to the same courses, and so they are indistinguishable. This forces us to consider a subset of the students as one unit, instead of multiple individuals. The theory applies the use of a formal approximation of a universe of discourse (crisp set or fuzzy set) through a lower and upper approximation of the original set [28]. In a universe of discourse A, the lower and upper approximation subsets are defined by $\underline{apr}(A)\, and\, \overline{apr}(A)$ respectively. The partition of a universe of discourse into two sets can be achieved by using constructive and algebraic methods. This research focuses on incorporating the impreciseness of sensor data into the map, and thus the use of rough sets is possible to meet this objective. However, if rough set theory is applied to incorporate sensor uncertainty information in robot environment maps, three sets $\underline{apr}(A)$, $\overline{apr}(A)$ and the rough boundary region set that represent values between lower and upper approximations should be defined and processed

[29]. This results in the definition of three rough membership functions complicating the representation of uncertainty in this research. On the other hand, only one membership function is required for incorporating uncertainty sensor information using fuzzy set theory. A detail comparison between rough sets and fuzzy sets can be found in [30]. For that reason, Fuzzy boundary representation is investigated for robot map representation in this thesis.

2.3.3. Dempster-Shafer Theory

It is the theory of belief functions. It is a generalization of the Bayesian theory of subjective probability. Whereas Bayesian theory requires probabilities for each question of interest, belief functions allow us to base degrees of belief for one question on probabilities for a related question. These degrees of belief may or may not have the mathematical properties of probabilities. How much they differ from probabilities will depend on how closely the two questions are related. Dempster-Shafer theory is based on two notions: the notion of obtaining degrees of belief for one question from subjective probabilities for a related question, and Dempster's rule for combining such degrees of belief when they are based on independent items of evidence [31]. To illustrate the use of Dempster-Shafer theory in robot map building, the degree of belief depends on sensors used to obtain the position of the obstacles (A Laser Range finder and an odometry sensor), and the two sensor measurements correspond to the items of evidence. Dempster-Shafer theory may be useful for deciding if there is really an obstacle at that position. However, in this thesis we propose to incorporate the sensor uncertainty information in the map. Using Dempster-Shafer theory only provides a degree of belief to the sensor measurement, this degree of belief is not adequate information when incorporating the sensor uncertainty in the map. Moreover, the belief functions are usually defined in the discrete domain *X*, and based on multinomial confidence regions. Complicated extension is required for application of Dempster-Shafer theory in continuous domains, like boundary representation of real robot environments [32] [33].

2.3.4. Fuzzy sets

Fuzzy sets are sets whose elements have graded degrees of membership. The use of fuzzy sets allows us to represent element membership in vaguely defined sets, which make it suitable for representing the concepts of vagueness and impreciseness. This graded degree of membership of the elements has the advantage over the sharp boundaries of the classical sets theory (members vs. non-members) in situations where there is impreciseness in the membership of elements data in the set. Fuzzy sets model impreciseness in a simple way and well-defined operations to combine information obtained by different sensor sources and modalities. Fuzzy sets are the most suitable way to represent the kind of uncertainty mentioned in section 2.3 where the prior information is the systematic error of the sensors, because, it allows us to model the uncertainty in one membership function and is easy to incorporate into the map. It is important to clarify that the objective of the research is not to reduce the uncertainty information, but to include such information into the map for decision-making. Interested readers may refer to Section 2.1 for introduction fuzzy sets.

2.4. Summary

In this chapter, background information was discussed. The chapter covered the basics of fuzzy set theory with common membership functions definition. Next, different robot map representations were classified in three categories (Raw sensor pattern, landmarks, and navigability of the space), their strengths, and limitations were discussed. Navigability of space map category was further classified in occupancy grids map, topological map, and boundary of navigable space. Details of strengths and limitations of each sub-category were given. Finally, different ways of representing uncertainty were discussed including probabilistic approaches, Rough set theory, Dempster-Shafer theory, and fuzzy set theory.

Chapter 3

Boundary Representation Mapping

The chapter details generation of robot environment map in the crisp world using B-rep systematically from sensor range data. The robot environment is considered static, in other words there is no moving object in the environment. Perfect range and odometry sensors are also assumed.

3.1 Boundary Representation

As previously mentioned on Chapter 2, Boundary Representation (B-rep) is a shape model that represents the environment by its boundaries of open area of robot environment, so it only takes in consideration boundaries of obstacles or walls [34]. The boundary representation model conveys both topological and geometrical information of the shape. For 3D objects, the geometric entities are faces, edges, and vertices. A face is an enclosed segment of several bounding curves or lines, edges are lines or curves segments between two vertices, and a vertex is the intersection point of adjacent edges.

The geometric information in B-rep of a shape corresponds to face, edge equations and vertex coordinates forming the basic components of a B-rep model. The topological information in B-rep of a shape corresponds to the spatial relationship among components, in other words how faces, edges and vertices are connected and holes (if any) in the shape [35]. Figure 3.1 shows the relation of different components of the B-rep of a 3D polygonal shape. A closed 3D polygonal object boundary representation contains at least four faces. A face contains at least three edges. An edge contains two vertices.

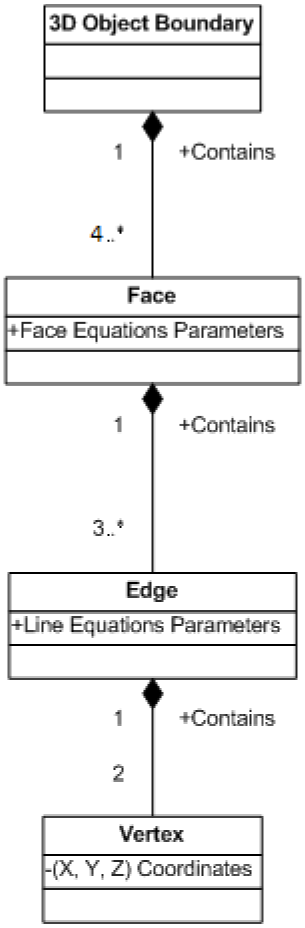

Figure 3.1 B-rep of a 3D polygonal object

3.1.1 2D Boundary representation

To represent 2D polygon shapes using B-rep, only edge and vertex information is required. The B-rep of a 2D polygonal shape concept is shown in Figure 3.2. A 2D polygon contains at least three edges and an edge contains exactly two vertices.

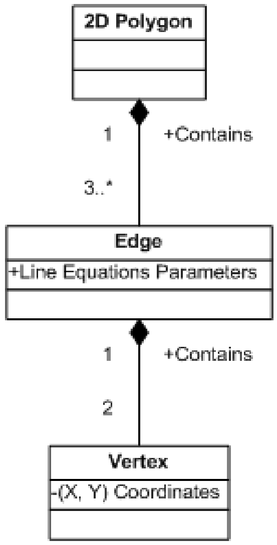

Figure 3.2 2D polygon in B-rep

The concept of a 2D Boundary representation of the robot environment is depicted in Figure 3.3. Overall, a B-rep environment map contains exactly one principal polygon and zero or more hole polygons for obstacles in robot environments. Principal and hole polygons are 2D boundaries represented by 2D polygons that have at least three edges and each edge is formed by joining two vertices.

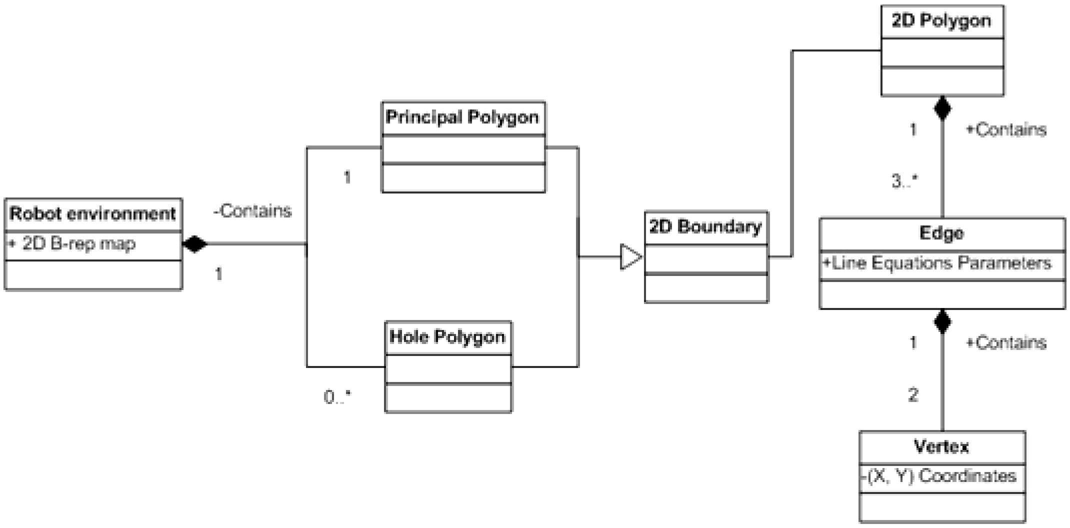

Figure 3.3 Robot environment in 2D B-rep map

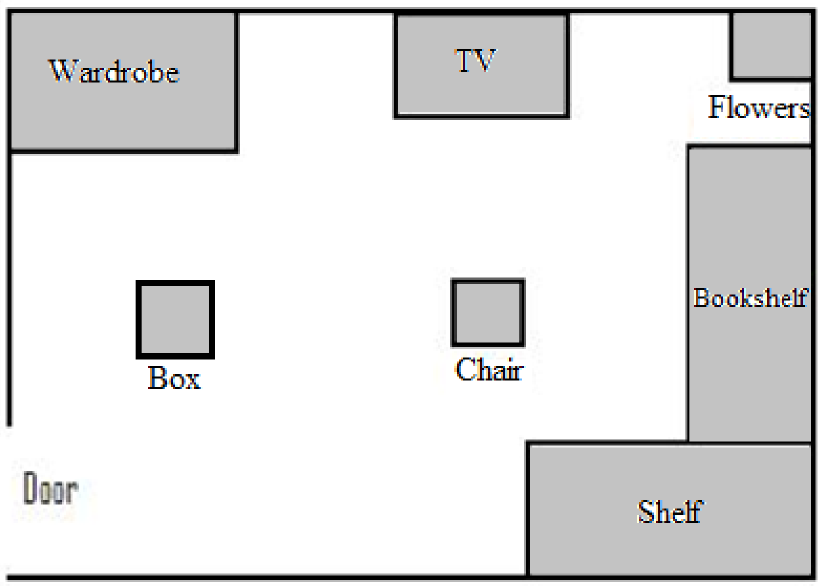

Figure 3.4 Office floor plan

Figure 3.4 shows the floor plan of a typical office for an autonomous household robot, where the white area represents free space and the shaded area is occupied space. The result of converting the office plan into a polygon like structure is depicted in Figure 3.5. The polygonal map contains 16 external vertices, 8 internal vertices, 16 external edges, and 8 internal edges. An obstacle in the physical environment is described as a hole in a B-rep map representation. Every hole in a B-rep map includes a vertex, and an edge table that describe it. Therefore, implementing a B-rep map to the office floor plan of Figure 3.4 results in three different boundaries data structures, one for the principal polygon (general floor description), two for the holes formed by the chair, and the box respectively.

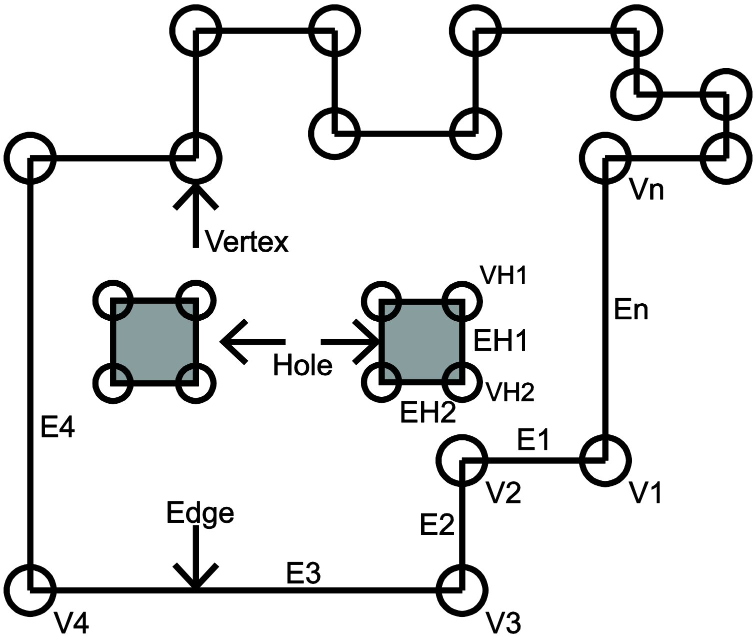

Figure 3.5 Boundary Representation of Office in Figure 3.4

3.1.2 Boundary representation map data structure

B-rep map data structure contains basic topological and geometric information of the robot environment. Topological information of the environment provides the relationships among its vertices, edges, and faces. Geometric information, on the other hand, refers to parameters of equations of the edges and faces [34] [36].

The wireframe model is a traditional way of representing 3D solids. It consists of two tables, namely the vertex table, and the edge table. A basic vertex table records vertices and its coordinates, while each entry in the edge table consists of two incident vertices of that edge. A wireframe model does not have face information [36]. Nevertheless, a face table may be included for better representation of complex 3D objects, but having face information to represent a 3D solid is not necessary (see [34] [36] for details).

Vertex V	X	Y
V1	X1	Y1
V2	X2	Y2
⋮	⋮	⋮
Vn	Xn	Yn

Table 3.1 2D Vertex table

Table 3.1 shows the basic 2D wireframe vertex table, and the only difference of the 3D vertex table is the lack of the Z coordinates of the vertex. Table 3.2 shows the basic structure of an edge table and extra columns of information can be added to include information of adjacent edges, line or curve model parameters, and so on.

Edge e	Vstart	Vend
E1	V1	V2
E2	V2	V3
⋮	⋮	⋮
En	Vn	V1

Table 3.2 Edge table

A hole polygon is represented in the same way as the principal polygon but in separate edge and vertex tables and each hole is individually indexed.

3.2 Generation of a boundary representation map from range data

To create the map of an unknown environment, the first step is to obtain the local scan of the area in the current position using sensors (Laser Range finder or ultrasonic sensor and odometry). A local scan, represented as a polygon, contains environmental information in the current pose (position and orientation of the robot) found by range data and the odometry sensors measurements. Secondly, that raw range sensor information (a collection of points representing obstacles) is converted into a closed polygon. A closed polygon is considered as the initial map of the area. Then the robot moves to a new location in free space of the environment and performs another local scan. Given the two polygons (initial map polygon and second scan polygon); the resultant map is obtained by taking union of both polygons to form one global polygon. Union of polygons may have holes in the result. This process is repeated by taking the local scan at a different pose and performing the union operation with the global polygon until the whole environment is scanned. Therefore, one global polygon (principal map) of the robot environment and smaller polygons inside if holes/obstacles are present is generated.

This thesis employs the Pioneer 3DX mobile robot as the experimental platform for building the B-rep map. It is equipped with a laser range finder and ultrasonic sensors. The laser range finder is employed as the default sensor since it has higher accuracy and angular resolution. Interested readers can refer to Chapter 5 for the specifications of the Pionner 3DX robot and the sensors installed.

The steps to create a 2D B-rep map are listed in the followings:

a) Obtain raw data from current robot pose. (Local scan)
b) Construct a scan polygon from the raw range sensor data. (The first polygon is the principal polygon, which is also the current global map).
c) Move the robot to a different pose. (Use a previously obtained polygon to move the robot to a free space. GVG [23] is a well-known method for path planning, this algorithm can be adapted to use the polygon as map in the robot exploration procedure)
d) Scan the area for new data. (Perform another local scan)

e) Construct a local scan polygon from the new data.

f) Perform union operation between the global map and the new polygon to consolidate the information into one principal polygon and hole polygon(s) (if any).

g) Repeat steps (c) to (f) until the whole robot environment is covered.

3.2.1 Construction of scan polygon from raw sensor data

The laser range finder LMS200 installed on the Pioneer 3DX robot is employed to acquire range data in robot environments. Figure 3.6 depicts a raw range data measurement that contains 181 points in which each point corresponds to a range measurement transformed to the Cartesian coordinate system (X, Y) from the polar coordinate system (ρ, θ). Once the raw range sensor data is obtained, the next step is to create a polygon from that data. RANSAC [37] [38] and Hough transform [39] are two common methods of extracting line segments from raw range data.

RANSAC is an iterative algorithm to estimate the parameters that fit a model from a set of data contaminated by outliers (data that do not fit the model). First, a hypothetical model is fitted to the inliers (data that fit the model). All data are then tested against the fitted model. The estimated model is considered acceptable if the numbers of members are higher than a preset CONSENSUS threshold. The model is re - estimated for other hypothetical inliers. Finally, the algorithm chooses the model that fit the best based on the CONSENSUS data and error tolerance. The RANSAC algorithm produces good results in some situations, but the application of the RANSAC algorithm to the small quantity of measures obtained through the LMS200 may produce under fitted results because the CONSENSUS parameter on the RANSAC algorithm should be small. This results in a tradeoff between the parameters CONSENSUS and TOLERANCE, and to obtain satisfactory line extraction results, the value of these two parameters need to adapt to different laser scans. On the other hand, Hough Transform algorithms can achieve accurate line equations extraction when the numbers of measurements is small according to our experimental experience. Therefore, this thesis employs Hough transform to extract the lines from the raw data.

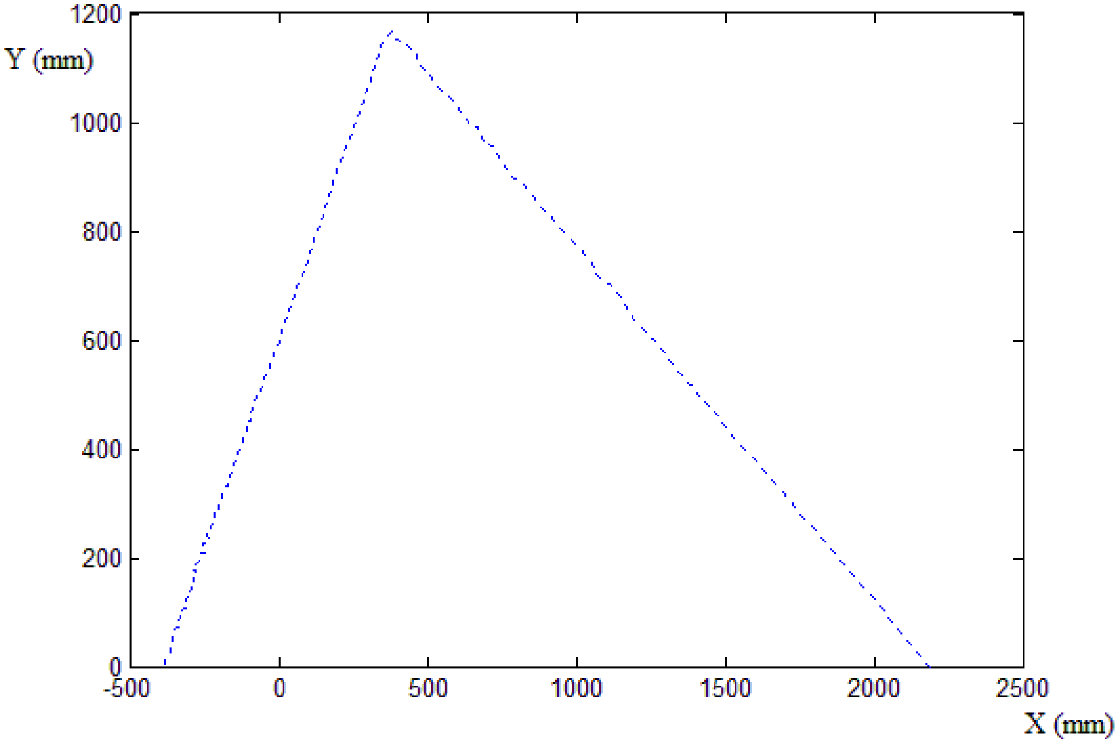

Figure 3.6 Laser range data of a corner

3.2.1.1 Hough Transform

Hough Transform [39] [40] is a general technique for identifying the location and orientation of certain types of features in digital images. It is possible to find all kinds of shapes that can be algebraically expressed (lines, circles, ellipses, etc.) using Hough Transform. In this thesis, lines are found from raw range sensor data. (Details on Hough Transform can be found in [39] [41] [42]).

In line extraction, the following equation $\rho(\theta) = xcos\theta + ysin\theta$ is used for the construction of the Hough Transform matrix, where x and y represent the Cartesian coordinates of one measured range point, ρ is the perpendicular distance between the origin and the line, θ is angle of the line being evaluated from the positive x-axis. By applying the equation for every $\theta \epsilon [0,180]$ in degree for all the range point measurements, the Hough Transform matrix or accumulator is constructed and is shown in Table 3.3 and Figure 3.7.

	$\theta = 0°$	$\theta = 1°$	…	$\theta = 180°$
Measurement#1 (X1,Y1)	$\rho_1(0)$	$\rho_1(1)$	…	$\rho_1(180)$
Measurement#2 (X2,Y2)	$\rho_2(0)$	$\rho_2(1)$	…	$\rho_2(180)$
⋮	⋮	⋮	⋮	⋮
Measurement#n (Xn,Yn)	$\rho_n(0)$	$\rho_n(1)$	…	$\rho_n(180)$

Table 3.3 Hough Transform Matrix or Accumulator

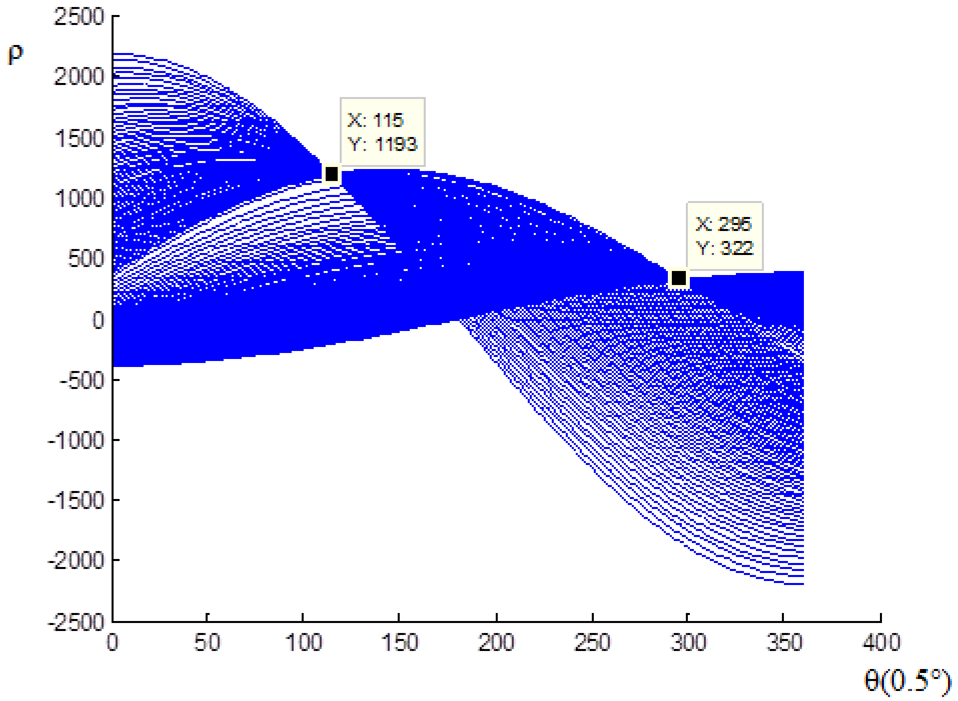

Figure 3.7 Graphical representation of the Hough Transform Accumulator of data in Table 3.3

Figure 3.7 shows the accumulator matrix generated by the measurements displayed in Figure 3.6. Note that the curves in the accumulator meet at two points. This means that there are two major lines reflected in the data shown in Figure 3.6.

After forming this matrix, it is required to determine how many points (measurements) belong to one line. Theoretically, two or more points fit in one line if they have the same $\rho(\theta)$ and θ. Tolerance is considered in comparing two $\rho_1(\theta)$ and $\rho_n(\theta)$ for line equation parameters estimation. The threshold of tolerance is set for minimum error ($\cong 1\%$ of initial $\rho_1(\theta)$ value). In case the line selection algorithm does not produce any results, the algorithm increases this tolerance and re-calculates the parameters of the fittest line. This algorithm is a vital part for the suitable selection of a line.

The result of this algorithm produces line equations parameters of lines constructed by the given range points. For example, by applying Hough Transform to Figure 3.6, two line equations: $y = -0.637x + 1410.270$ and $y = +1.570x + 608.100$ are generated as shown in Figure 3.8.

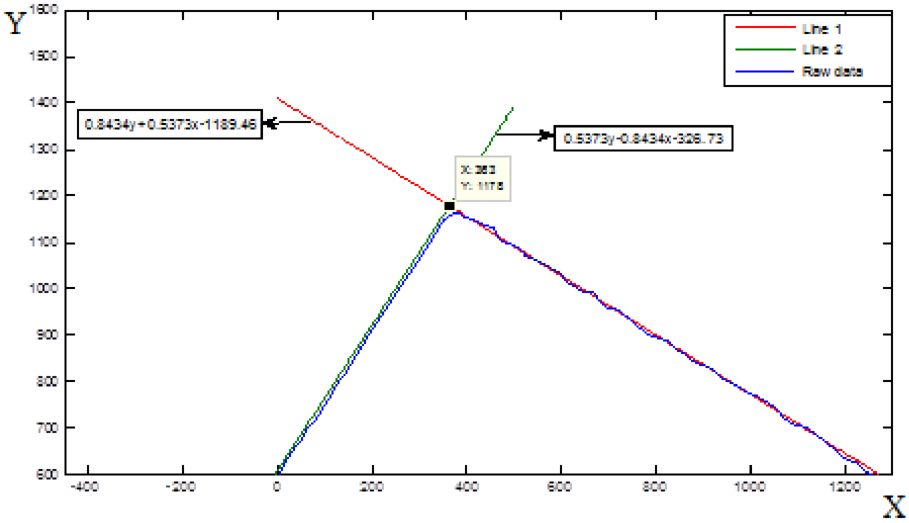

Figure 3.8 Resulting lines obtained from Hough Transform and their intersection

3.2.1.2 Finding the vertex points

The Hough Transform decision maker algorithm generates the equation parameters (ρ and θ) of lines fitted by range data points acquired in a single laser range scan. Since the range data points are ordered along the angular dimension (from 0°, 1°, 2°, to 180°), the extracted lines can be ordered according to the indices of the range data points that are fitted to them. By exploiting this fact, computation of the intersection points can be greatly reduced, given that the intersection points are computed one by one according to the extracted line order, instead of finding intersection points from a collection of unsorted lines. The first extracted line has one intersection with the second line; the second line has one intersection with the third line and so on.

Given two line equations, the intersection point is given as:

$$(1) \quad Ay + Bx + C = 0 \rightarrow y = \frac{-Bx - C}{A}$$

$$(2) \quad Dy + Ex + F = 0 \rightarrow y = \frac{-Ex - F}{D}$$

$$(3) \quad x = \frac{CD - AF}{AE - BD}$$

Algorithm is also developed to handle special case when AE=BD, when one of the lines is either vertical or horizontal.

The coordinates of all vertices (intersection points) of the scan polygon are generated and line equation parameters of all edges (lines) of the scan polygon are obtained using Hough Transform. The vertex table then contains the coordinates of all the intersection points taken as vertices as shown in Table 3.1. The edge table contains the vertices information as shown in Table 3.2.

3.2.2 Polygon Union in Boundary Representation

This section discusses how to perform union operation on two polygons in B-rep data structure. Each polygon data includes one vertex table and one edge table as shown in Table 3.1 and 3.2. In Figure 3.9, two polygons overlap. The first step is to find the vertex points of the union polygon (displayed as red point in Figure 3.9). This is achieved by analyzing every edge line equation of one polygon with the second polygon in search for the intersection point of the edges. If an intersection occurs within the line segment (indicated by the vertex points of the evaluated edges), the new intersection point is considered as the new vertex point of the union of polygon.

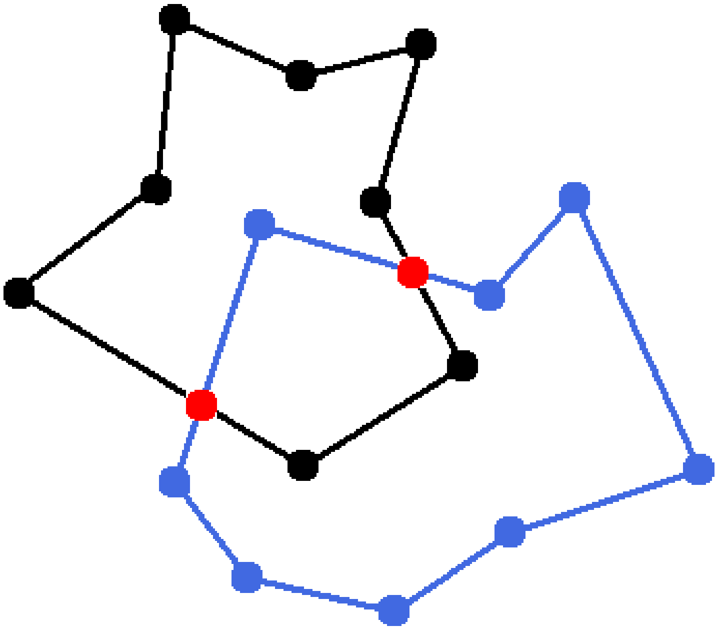

Figure 3.9 Union of 2D polygons

Assume the black polygon is polygon A and the blue polygon is polygon B. One polygon is the B-rep current global map (polygon A), and the polygon B is the new local scan polygon obtained by new laser scan. The edges of each polygon are ordered in a counter-clockwise manner. Both polygons A and B have eight edges. With the introduction of new vertices (red dots in Figure 3.9), both polygons now have 10 edges as shown in Figure 3.10.

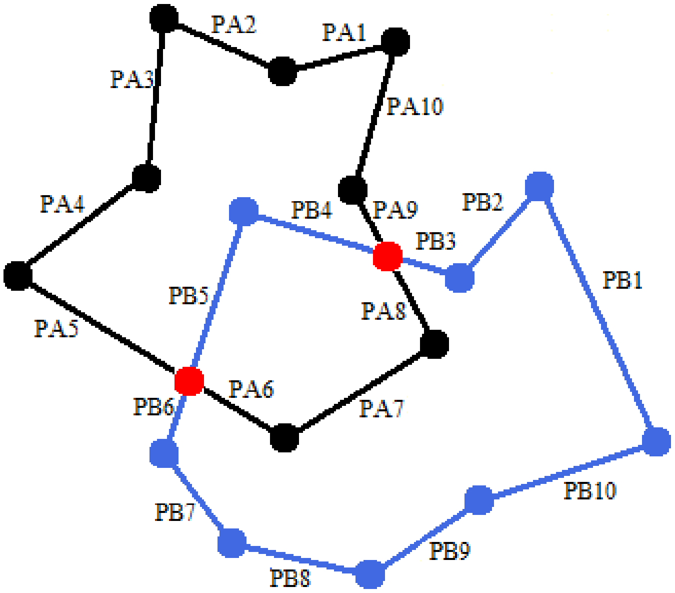

Figure 3.10 Union of 2D polygons (identified edges with the new additions)

The result of the union operation just considers edges that exist outside both polygons. Straightforwardly, the union operation scans each edge and decides if the edge is “INSIDE” or “OUTSIDE” of the other polygon. Initially the scan start with the first edge of polygon A (PA1), pick a point arbitrarily that lies within the first edge. Its coordinates can be easily obtained from the parametric form of the line equation of this edge. This point is called evaluation point and is used to determine if the edge can be classified as “OUTSIDE” or “INSIDE” a given polygon according to algorithm 1 developed by Randolph [43].

Algorithm 1 .

```
Input
    Testy= Y-coordinate of test point.
    Testx= X-coordinate of test point.
    V1y and V2y=Y-coordinate of current edge
    V1x and V2x=X-coordinate of current edge
    C= returned value.
    C←False; “(False means “OUTSIDE,” True means “INSIDE”)
    For each edge of the polygon
        If ((V1y>Texty) and (V2y<Testy) or (V2y>Testy) and (V1y<Testy))
            If ( $Testx < \frac{(\mathrm{V1x}-\mathrm{V2x})*(\mathrm{Testy}-\mathrm{V2y})}{((V1y-V2y)+V2x)}$ )
                C=not C;
            End;
        End;
    End.
Output C;
                                                                    end.
```

For the point to be inside the polygon the condition has to be met for an odd number of edges because the initial value of C is false and C is negated every time the condition is met.

After classifying every edge of both polygons as “OUTSIDE” or “INSIDE,” the next step is to scan along the edges for all the “OUTSIDE” edges that belong to the union polygon.

For example, in Figure 3.10 start with edge PA1 in polygon A. If the edge is labeled as “OUTSIDE” polygon B then it belongs to the result of the union operation. Repeat the check for the next edges PA2, PA3 up to PA6 where this edge is classified as “INSIDE.” That means that

PA6 does not belong to the union polygon. At this point, the scan meets an edge that is “INSIDE” of polygon B. The algorithm then needs to compare for equality between the end vertex point (Vend) of the previous edge (PA5 Vend) with the Vstart (See Table 3.2) of the all the edges of polygon B. The scan continues from edge where the Vstart of polygon B is equal to the Vend of the last “OUTSIDE” edge of polygon A (PA5). After switching to the second polygon (polygon B), continue to scan along the edges in the counter-clockwise manner up to PB4 that is labeled as “INSIDE” of polygon A, and once again this “INSIDE” edge does not belong to the resulting union polygon. The algorithm performs the same check, as previously, when an “INSIDE” edge was encountered and transfer the scan to the other polygon (polygon A) from the edge where the Vstart of polygon A is equal to the Vend of the last “OUTSIDE” edge of polygon B. The switching edge now becomes PA9. The scan continues to check for “OUTSIDE” edges from PA9 and go back to PA1 that has already been scanned. This marks the end of the scan and a new closed polygon is generated.

The union polygon is shown in Figure 3.11 where edges PB4, PB5, PA6, PA7, and PA8 are removed from the resulting polygon. The last step is to rename the edges of the resulting polygon to form one global map. This algorithm performs the union operation between two convex polygons so that no hole is formed after the union operation.

This algorithm for “INSIDE/OUTSIDE” edge scanning focuses on the polygon switching logic and assumes that the two polygons overlap. If the polygons do not overlap (intersection points between the two polygons less than 2) the union operation is straightforward. Both polygons belong to their union.

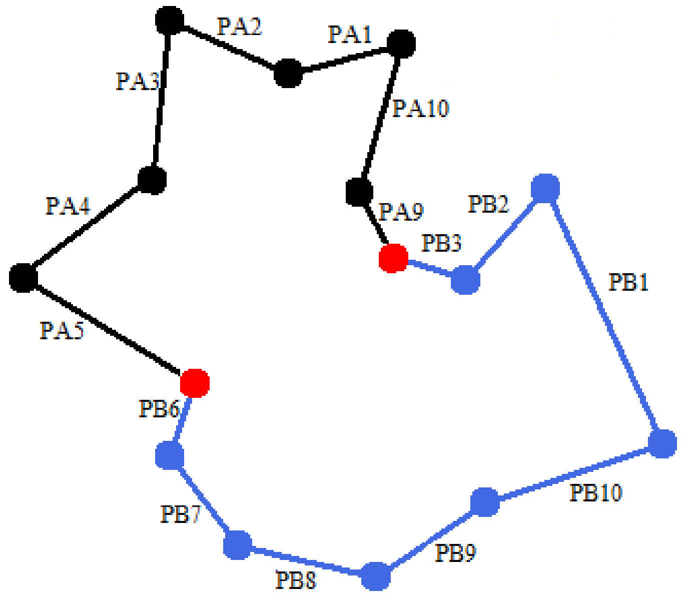

Figure 3.11 Resulting Union Operation of the Polygon A and B

3.2.3 Management of holes in union

In previous section, the union operation between two convex polygons was explained. However, in some cases the union operation of two or more concave polygons may introduce holes in the union polygon.

This section describes how to detect and handle holes in the B-rep map structure when union of the polygons is formed. A hole in the map represents objects occupied in the robot environment, which may be a static object or a moving one.

The question arises, "How a **hole** is detected in the resulting polygon from the union operation?"

The union operation algorithm classifies all the edges of both polygons as "INSIDE" or "OUTSIDE" in relation with the other polygon and that all the edges classified as "OUTSIDE" are part of the final union result. Before starting to check for "OUTSIDE" edges, flag all the edges as "NOT SCANNED," which means that the edge has not been checked if it belongs to the final solution or not. Once the scan begins, toggle the flag of the checked edges to "SCANNED." After all the edges are checked, verify if there are still some edges that are classified as "OUTSIDE" and flagged "NOT SCANNED." This means that some "OUTSIDE" edges are not part of the final solution. This implies that a hole may exist in the resulting union polygon. Some of the conditions to determine if the resulting polygon has a hole or not are defined.

- All edges in holes are also classified as "OUTSIDE" in the union algorithm.
- All edges that belong to a hole have to be flagged as "NOT SCANNED."
- There has to be a minimum of three edges classified as "OUTSIDE" and flagged "NOT SCANNED." A hole is also a closed polygon, and thus it must have a minimum of three edges.
- If the previous conditions are true then start a new scan with only the edges classified as "OUTSIDE" and flagged "NOT SCANNED." If this scan forms a new closed polygon then there is a hole in the union result.
- Inspect for another hole, until all edges are exhausted.

In order to demonstrate the steps clearly, an example of the union operation is given in Figure 3.12 and the polygons have already been handled for the union operation (intersections and new edges added).

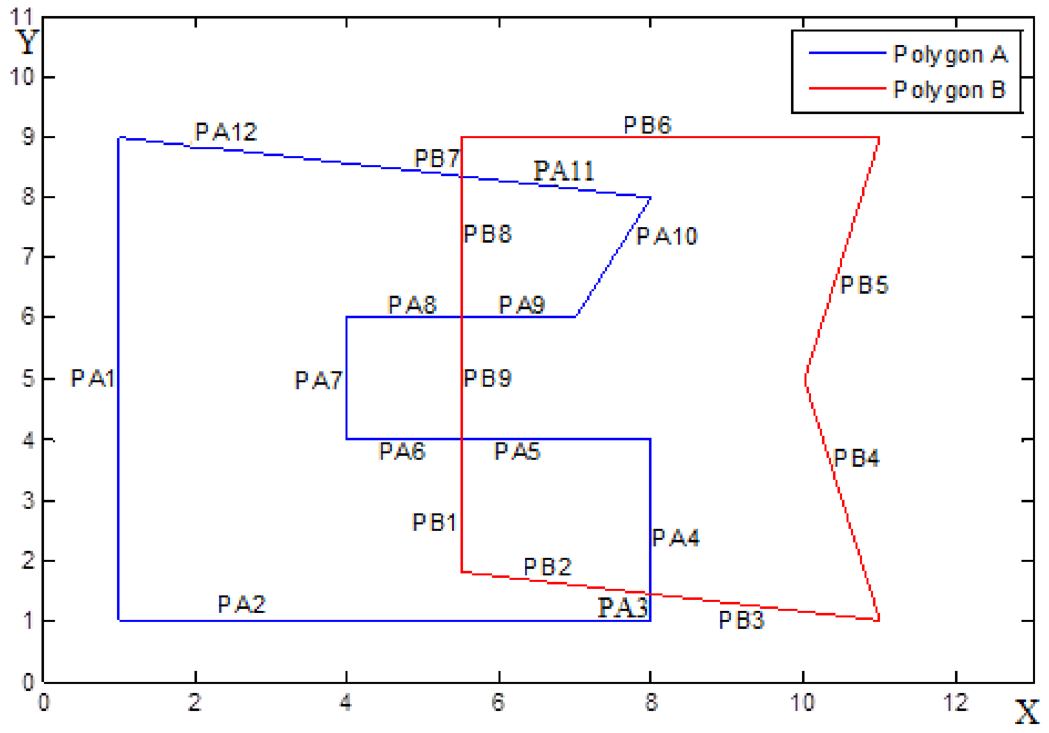

Figure 3.12 Union operation of two polygons with a resulting hole

The edges of both polygons are classified as "INSIDE" or "OUTSIDE" of the other polygon and the result is listed in Table 3.4 (all edges are currently flagged as "NOT SCANNED").

Polygon A	Classification		Polygon B	Classification
PA1	Outside		PB1	Inside
PA2	Outside		PB2	Inside
PA3	Outside		PB3	Outside
PA4	Inside		PB4	Outside
PA5	Inside		PB5	Outside
PA6	Outside		PB6	Outside
PA7	Outside		PB7	Outside
PA8	Outside		PB8	Inside
PA9	Inside		PB9	Outside
PA10	Inside			
PA11	Inside			
PA12	Outside			

Table 3.4 Edge classification of polygon A and B

Subsequently, start the first scan for all edges of the polygon and flag inspected edges as “SCANNED,” so that the principal boundary of the union polygon is obtained. The edges classification and flag after the first scan is listed in Table 3.5. Since some edges are flagged as “NOT SCANNED” and classified as “OUTSIDE,” hole(s) is/are formed in the union polygon. As shown in Table 3.5, four edges are classified, as “OUTSIDE” and they are PA6, PA7, PA8, and PB9.

Polygon A	Classification	Flag		Polygon B	Classification	Flag
PA1	Outside	SCANNED		PB1	Inside	NOT SCANNED
PA2	Outside	SCANNED		PB2	Inside	NOT SCANNED
PA3	Outside	SCANNED		PB3	Outside	SCANNED
PA4	Inside	NOT SCANNED		PB4	Outside	SCANNED
PA5	Inside	NOT SCANNED		PB5	Outside	SCANNED
PA6	Outside	NOT SCANNED		PB6	Outside	SCANNED
PA7	Outside	NOT SCANNED		PB7	Outside	SCANNED
PA8	Outside	NOT SCANNED		PB8	Inside	NOT SCANNED
PA9	Inside	NOT SCANNED		PB9	Outside	NOT SCANNED
PA10	Inside	NOT SCANNED				
PA11	Inside	NOT SCANNED				
PA12	Outside	SCANNED				

Table 3.5 Edge classification and flag of polygon A and B after first scan

The final step for the management of holes is to start the second scan from one of the edges flagged as “NOT SCANNED” and classified as “OUTSIDE” to find the hole polygon. For example, in Figure 3.10 start the scan from edge PA6, following the rules (described in the union operation algorithm) PA6 is part of the final solution because it is an “OUTSIDE” edge, next PA7 and PA8 follows and are also considered as part of the hole. Then, PA9 follows, however this edge is classified as “INSIDE” thus the search of the equality of the vertices on the other polygon is initiated. The search then chooses PB9 as the next edge to be inspected and make it part of the hole. It is because it is classified as “OUTSIDE” and flagged “NOT SCANNED.” Next, PB1 is labeled as an “INSIDE” edge thus the search for equality of the vertices on the other polygon is initiated. PA6 is then selected as the next edge to be inspected. However, PA6 is already part of the final solution and is flagged as “SCANNED” and thus it completes the second scan with a closed polygon representing a hole. In the case, there still are edges classified as “OUTSIDE” and flagged “NOT SCANNED,” a new scan in search for another possible hole is launched.

Figure 3.13 shows the result of the union operation of polygon A and B shown in Figure 3.12. The Table 3.6 and 3.7 list the information of the principal boundary and the hole boundary respectively of the result in permanent storage.

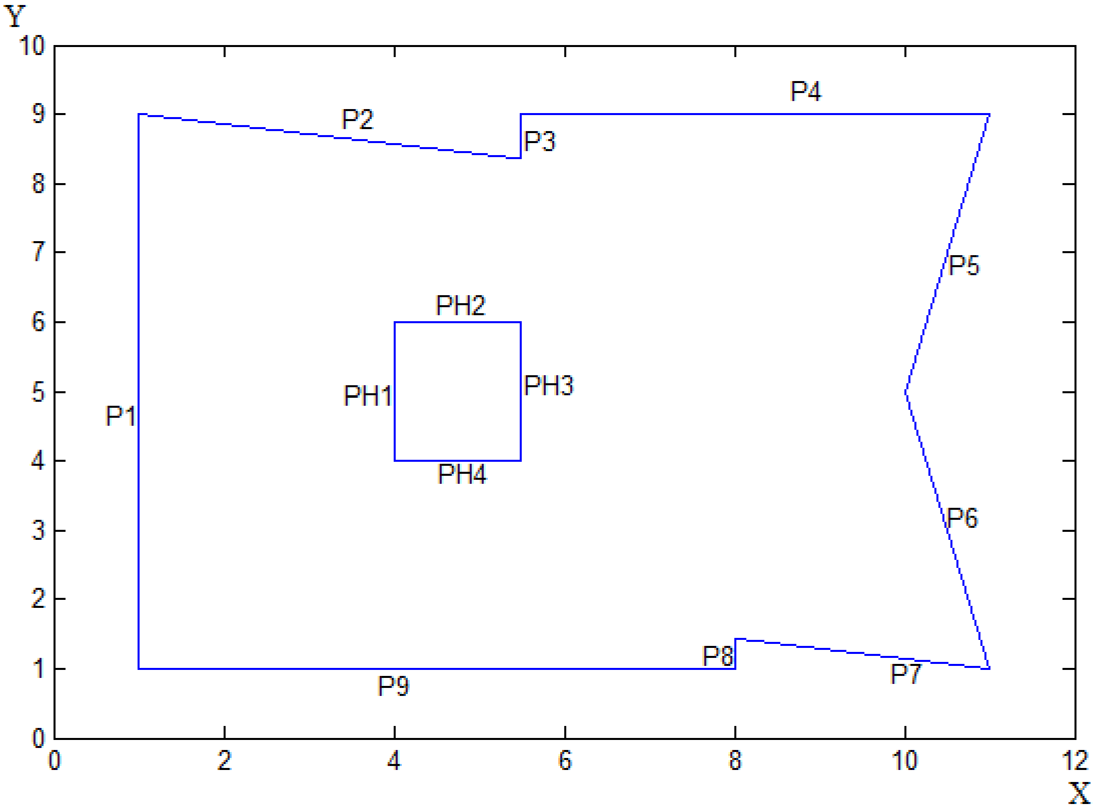

Figure 3.13 Result of the Union operation from polygon A and B of Fig 3.12

Edge	Vstart	Vend		Vertex	X	Y
P1	VP1	VP2		VP1	XP1	YP1
P2	VP2	VP3		VP2	XP2	YP2
P3	VP3	VP4		VP3	XP3	YP3
P4	VP4	VP5		VP4	XP4	YP4
P5	VP5	VP6		VP5	XP5	YP5
P6	VP6	VP7		VP6	XP6	YP6
P7	VP7	VP8		VP7	XP7	YP7
P8	VP8	VP9		VP8	XP8	YP8
P9	VP9	VP1		VP9	XP9	YP9

Table 3.6 Edge table and Vertex table for main polygon of Fig 3.13

Edge HOLE	Vstart	Vend		Vertex	X	Y
PH1	VH1	VH2		VH1	XH1	YH1
PH2	VH2	VH3		VH2	XH2	YH2
PH3	VH3	VH4		VH3	XH3	YH3
PH4	VH4	VH1		VH4	XH4	YH4

Table 3.7 Edge table and Vertex table for polygon representing the hole in Fig 3.13

3.3 Summary

In this chapter, the generation of a boundary representation map, its data structure, and the management of holes were introduced. Moreover, the chapter described an algorithm for generation of B-rep map from raw range data. It also described essential operations needed for formation of B-rep map from local scans, like the union operation between two different polygons and the management of objects (holes) in the area (map).

Chapter 4

Fuzzy Boundary Representation

In chapter 3, a B-rep map of the robot environment in the crisp world was constructed. However, the crisp B-rep map does not take into consideration the uncertainty information of the sensors (laser range finder and odometry sensors). Imperfect sensors are the principal source of uncertainty in the creation of a map; this is because the data obtained through them is not completely precise. This impreciseness in sensor data cannot imply the exact location of features or objects in the map. This chapter introduces the incorporation of uncertainty information into the robot map by fuzzifying the B-rep map. The chapter presents the fundamental steps to convert the crisp B-rep map to its fuzzy counterpart. These include the selection of a sensor uncertainty membership function for uncertainty modeling, the sweeping operation for α-cut generation, and the data structure for fuzzy boundary representation (fuzzy B-rep).

4.1. From the crisp world to the fuzzy world

The conversion from a universe of discourse to its fuzzy counterpart (grades of membership) is called fuzzification.

The main step of fuzzifying any concept or process is to generate the fuzzy membership function. The membership function is used to associate a grade membership value to the crisp data. In robot map building using boundary representation, all boundaries defined in a fuzzy B-rep become an edge defined by a membership function governed by the sensor uncertainty.

Therefore, selecting the appropriate membership function to reflect sensor uncertainty information for every edge in the B-rep robot map is important. This membership function is called sensor uncertainty membership function (fuzzy set). The membership function of the fuzzy boundary representation is constructed by grouping all of its α-cuts. After the selection of this sensor uncertainty membership function, the sweeping operation is performed to generate the α-cuts of the boundary of robot environment. The sweeping operation consists of passing a "moving entity" (sensor uncertainty membership function) along a "trajectory" given by the crisp B-rep polygon. This "moving object" passes through all the edges of the crisp B-rep polygon to generate the fuzzy B-rep map. This sweeping operation is similar to the sweep representation of a solid [34]. Figure 4.1 illustrates a simple fuzzy B-rep polygon, its corresponding inward and outward offsets produced by the support (μ=0) of the membership function, and one α-cut at μ=0.4013.

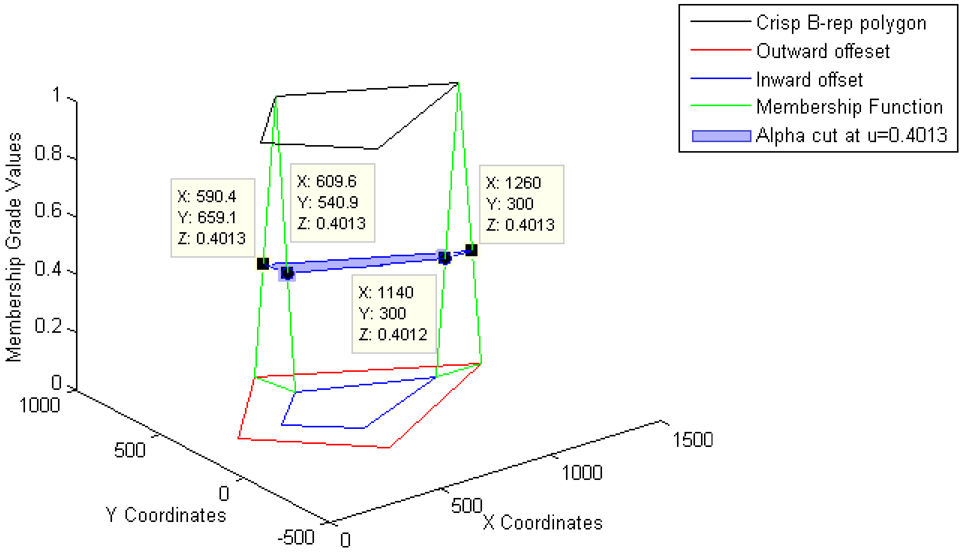

Figure 4.1 3D view of a Fuzzy B-rep polygon

Figure 4.2 illustrate a different perspective of Figure 4.1. Figure 4.2 shows the crisp B-rep polygon that corresponds the core (μ=1), the support (μ=0) and the polygons at μ=0.8, μ=0.6, μ=0.4, and μ=0.2, generated by sweeping a triangular sensor uncertainty fuzzy set along the crisp B-rep polygon.

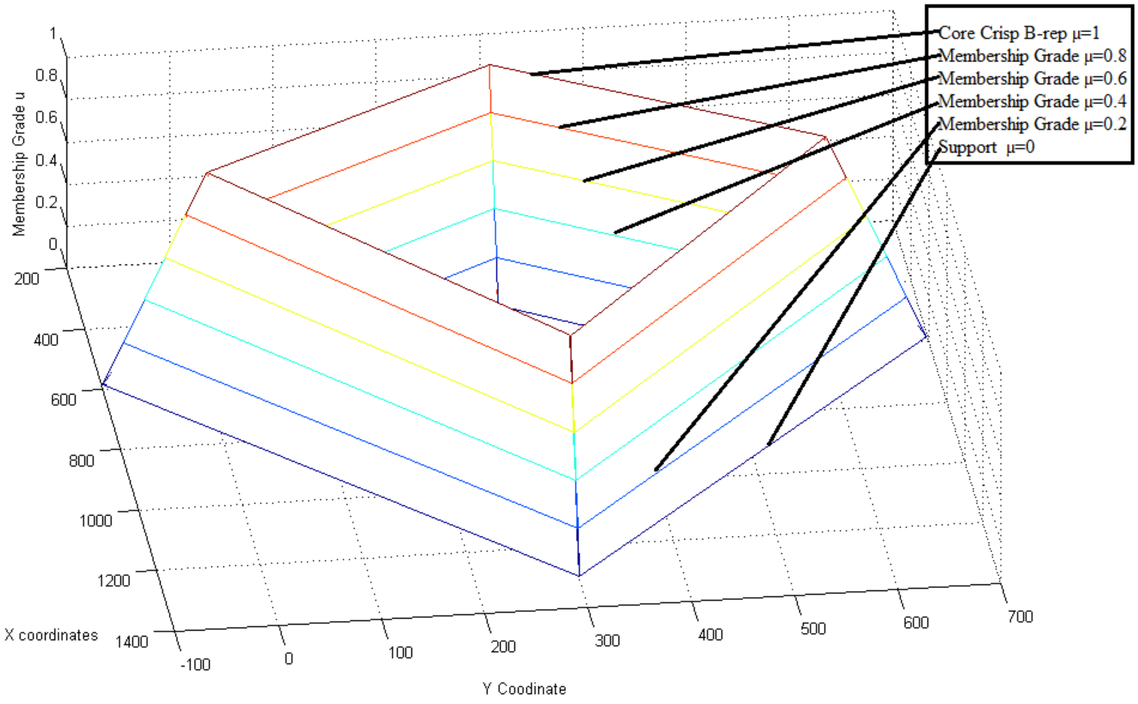

Figure 4.2 3D view of a Fuzzy B-rep polygon in Figure 4.1

4.1.1 Selection of a proper sensor uncertainty membership function

As mentioned previously, the sensors uncertainty is the principal source of uncertainty in the obtained scan polygons. This means that when constructing the sensor uncertainty membership function the errors introduced by the sensors have to be taken into consideration. The sensors involved in this research are:

- Laser range finder (LRF)
- Odometry

The measurement error inherent in the laser range finder LMS200 is given by the technical specification of the sensor [44]. This sensor has a range of 8 meters and has a systematic error of ±15mm on normal conditions (See Chapter 5 for its specification). On the other hand, the measurement error introduced by the odometry sensor is not given by its technical specification but it can be obtained through calibration experiments using the UMBmark method [45](refer to Appendix I).

The main step of the fuzzification of B-rep maps is to characterize the uncertainty introduced by the sensors to create the membership function. Both sensors errors need to be considered to create a single possible maximum error and incorporate all sensors uncertainty into the fuzzy B-rep map. Thus, selecting the maximum error as support for the sensor uncertainty membership function is essential.

Sensor measurement error values:

$LRF\ error = \pm 15mm$ (Given by its technical specification [44])

$Odometry\ error = \pm 184.13mm$ (Calculation in Appendix I)

$TME = \pm |abs(LRF\ error) + abs(Odometry\ error)| = \pm 199.13mm$

The addition of both sensor errors assured that the calculated TME (Total Maximum Error) value is the maximum possible error. This value specified the safety margin for navigation and path planning.

As mentioned previously in section 2.1.1, the selection of the sensor uncertainty membership function is highly related to the nature and the uncertainties of the process. A boundary representation map represents the boundaries of an environment. However, the boundary obtained may not be exact at the measured position. Fuzzification of a crisp B-rep map is introduced to model the vagueness of the boundaries position obtained through sensors.

The sensor uncertainty membership function grants a graded value to a position data, and this graded value represents the uncertainty of whether that position is open or not. To illustrate, a real number **x** representing location with a membership grade value of $\mu_x = 1$ means that the location **x** is considered a boundary, and so a membership value $\mu_x = 0$ would mean that the position **x** is not a boundary at all. Therefore, three criteria for defining the sensor uncertainty membership function are proposed:

1. $\mu_x = 0 \; if \; x \geq (r + TME)$,
2. $\mu_x = 0 \; if \; x \leq (r - TME)$ and
3. $\mu_x = 1 \; if \; x = r$

Where **x** is the real number to be evaluated, **TME** is the total maximum error (that depends on sensor measurement errors specification) and **r** is the value where its membership is equal to one. These criteria state that everything that is outside the range defined in criteria #1 and #2 has a membership grade value equal to zero. In other words, any position **x** outside that range is not considered a boundary. Criterion #3 states that the original measure **r** is considered a boundary (obtained in the crisp world in Chapter 3). These criteria put constraints in the selection of the sensor uncertainty membership function. Any function that meets these three criteria can be applied to describe the impreciseness of the sensor measurements (Figure 2.3 shows some membership function examples).

A question arises “What does the membership function imply?” To answer this question two different membership function shapes are given as examples. Figure 4.3 shows a membership function with a sharp peak. This kind of membership function implies that the data taken by the sensors are highly accurate, because the area under the graph of membership function $\mu_x > 0$ is small, implying that the membership grade values of the data representing position are small, thus a smaller area in the environment is considered a boundary.

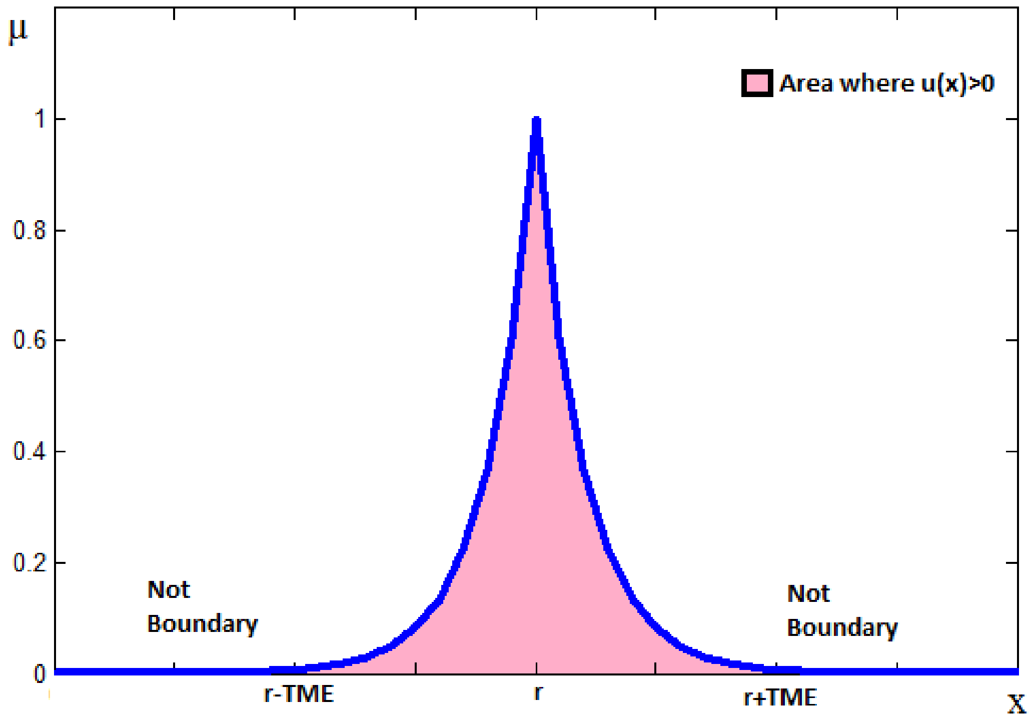

Figure 4.3 Membership Function with a pointed peak

On the contrary, a bigger area under the function implies that the sensor uncertainty is high. Figure 4.4 shows a function where the area of the region under the curve of the membership function is big. Positions covered by the pink shaded area have a high membership grade value that characterizes high impreciseness of the position information of the boundary.

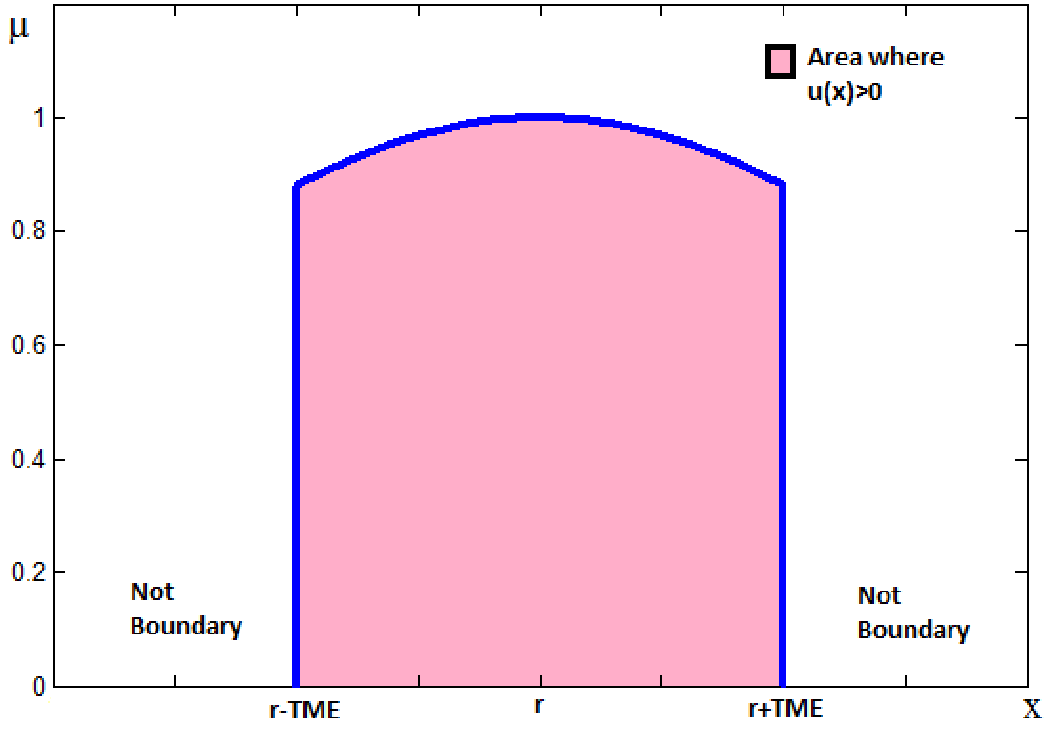

Figure 4.4 Membership function with high uncertainty

In other words, the bigger the area under the sensor uncertainty membership functions, the higher the uncertainty in sensor measurements. In the worst-case scenario, all measured range X that lies between $r - TME\ and\ r + TME$ has a membership grade value equal to one, and the whole area is considered a boundary. Accordingly, the best-case scenario is that the sensor measurements are error free and all measured range excluding the original measurement **r** has a membership grade of zero. This reduces the fuzzy boundary to its crisp counterpart (See Chapter 3).

However, without prior knowledge of the material, of which the environment is made, finding the shape of the sensor uncertainty membership function is difficult. A triangular sensor uncertainty membership function is reasonable choice overall, because it corresponds to the average cases across the preciseness and impreciseness continuum. The triangular membership function is as follows.

- $a = r - TME;$
- $b = r + TME;$
- $\mu_x = \frac{(x-a)}{(r-a)};\ when\ x \in [r - TME, r)$
- $\mu_x = \frac{(b-x)}{(b-r)};\ when\ x \in [r, r + TME]$

As shown in Figure 4.5, the width of the support of the triangular membership function is twice of TME, where **a** and **b** are the lower and upper limits of the support of the membership function respectively.

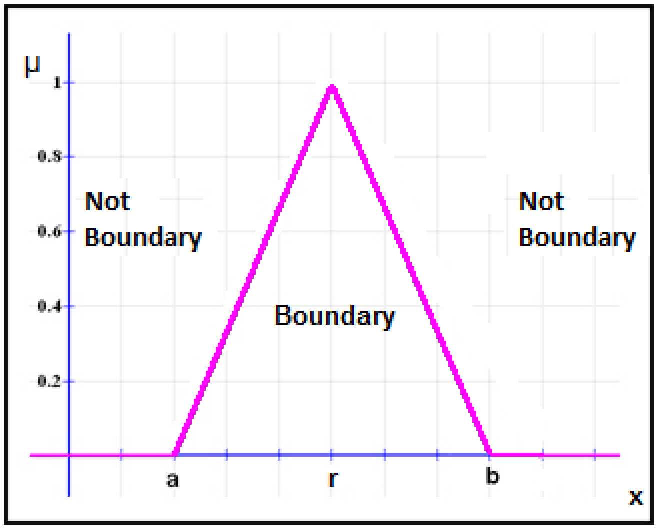

Figure 4.5 Triangular Membership function

Nevertheless, the membership function is not required to have only one shape and is subject to change, if the robot can recognize that the measured area is highly certain or uncertain for the sensor in use. If a laser range finder is used for measurement, an area where the reflectivity of the laser is low (e.g., glass fences), the range measurement data can be considered as uncertain. On the other hand, the range data measurement is certain when the area is made of material with high laser reflectivity (e.g., opaque walls). Thus, the sensor technical specifications and the material of which the environment is made determine the shape of sensor uncertainty membership function. In this thesis, the triangular membership function is used. The flexibility of using different membership functions is presented and taken into consideration in the fuzzy B-rep map data structure (see Chapter 4.3, table 4.1).

4.1.2 α-cut generation by sweeping operation

Once the sensor uncertainty membership function is determined, α-cut generation of fuzzy boundaries is made by sweeping the sensor uncertainty fuzzy set along the crisp boundary representation of the robot map generated from range data (Chapter 3). Polygon offsetting provides a simple solution to finding the lower and upper bounds of the α-cut of the fuzzy boundary membership function. As shown on Figure 4.6 offsetting tries to keep the shape of the original polygon. Polygon offsetting can be easily implemented in a B-rep data structure. The lower and upper bounds of the α-cut of sensor uncertainty membership function shown in Figure 4.1 define the offsetting distance. An inward and outward offsetting is executed with the lower and upper limits defined by the α-cut of the sensor uncertainty membership function.

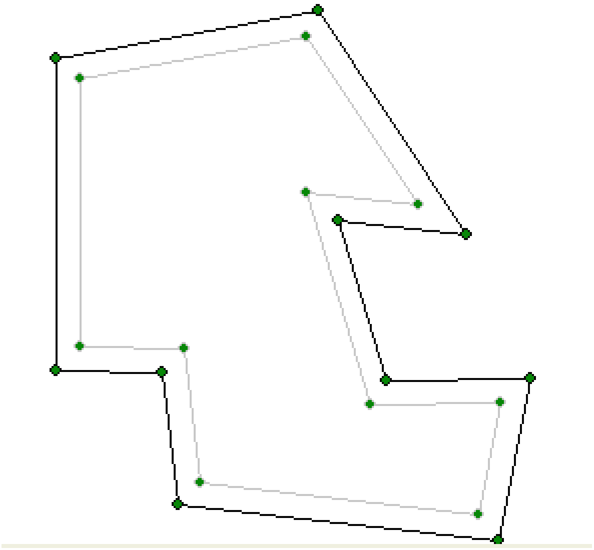

Figure 4.6 Polygon offsetting

4.1.2.1 Offsetting operation

There are many well-known algorithms for offsetting a polygon. Wein's method [46] uses the Minkowski sum [47] with a disc of radius **R** to offset the polygon. The author also included an exact offsetting algorithm using algebraic number types and an approximation algorithm that use rational arithmetic to reduce computational time. On the other hand, Kim [48] developed the multiple normal vectors algorithm for offsetting a 3D triangular mesh. The author also introduced the use of a single normal vertex, but the accuracy is poor in 3D offsetting [48] [49]. By combining the attributes of single normal vertex and rudimentary algebraic computation used in the Wein approximation algorithm [46], a new hybrid algorithm for precise offsetting of a 2D polygon is proposed. The use of the single normal vertex method has the advantage of not creating gaps in the resulting polygon, and lower computational complexity than other methods discussed in [48]. This method tries to keep the exact shape as the original polygon. Another advantage for using single normal vertex method is that the information required to carry out this algorithm can be obtained directly from the B-rep data structure.

The proposed algorithm applies algebraic computation to find the single normal vertex line equation. This equation is used for easy offsetting with varying offsetting distance, which is given by the lower and upper bounds of the α-cut of the sensor uncertainty membership function at a particular membership grade (See Figure 4.1, and Figure 4.2). In other words, to find the lower and upper bounds of an α-cut for the given fuzzy boundary membership function requires only computing the new vertex points along the normal vertex line. This is the main reason for using the proposed offsetting algorithm instead of previously developed ones [46] [48] [49], where finding an α-cut of the membership function requires applying the whole offsetting algorithm for different radius or distance specified by the lower and upper bounds of the α-cut in the sensor uncertainty membership function.

The B-rep map data structure contains the edge table and vertex table as specified in Chapter 3. Obviously, a corner is produced by two linked edges with one vertex point in common as shown in Figure 4.7. V2 is the vertex point of the intersection between edges ED1 and ED2. Figure 4.7 also shows the single normal vertex point (V2N) that lies on the angle bisector the line (V2-V2N) of the angle $\angle V1V2V3$.

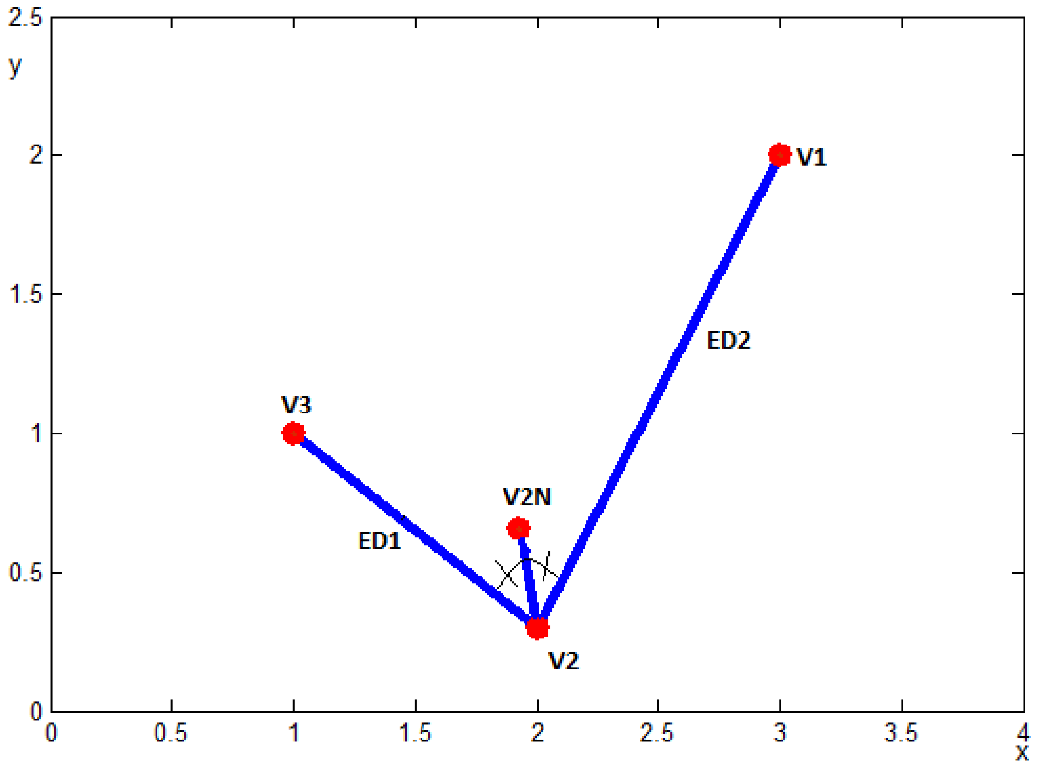

Figure 4.7 corner with the single normal vertex of V2

For symmetric sensor uncertainty membership function, V2N is found at a distance of half the range of the α-cut of the sensor uncertainty fuzzy set at a particular membership grade from the vertex V2. For example, at membership grade μ=0, the distance corresponds to half the support of the fuzzy membership function. It is worth to mention that the V2N vertex point on Figure 4.7 represents the outward offsetting vertex point of the V2 point. When the line equation is found and converted to the parametric form, the location of the inward offsetting vertex point of V2 can be extrapolated by accordingly changing the sign of the parameter in the parametric line equation.

Once the radial position of V2N is determined, the algorithm needs to find the angular position of V2N, β, which is formed between the first and second edges of the intersection in a counterclockwise direction as shown in different cases of β in Figure 4.7.

As seen in Figure 4.8 the computation of the angle β depends on the position of the vertices V1 and V3. In Figure 4.8 A, V1 is located in the first quadrant and V3 on the adjacent quadrant of the Cartesian plane. In Figure 4.8 B, V1 and V3 are reversed compared with Figure 4.8 A. In Figure 4.8 C and D, both vertices V1 and V3 are found in the same quadrant, and their positions are reversed in Figure 4.8 C and D respectively. In Figure 4.8 E and F, both vertices V1 and V3 are found in opposite quadrants, and their positions are reversed in Figure 4.8 E and F respectively. The orientation of the edges and the vertices is important for the calculation of the angle β.

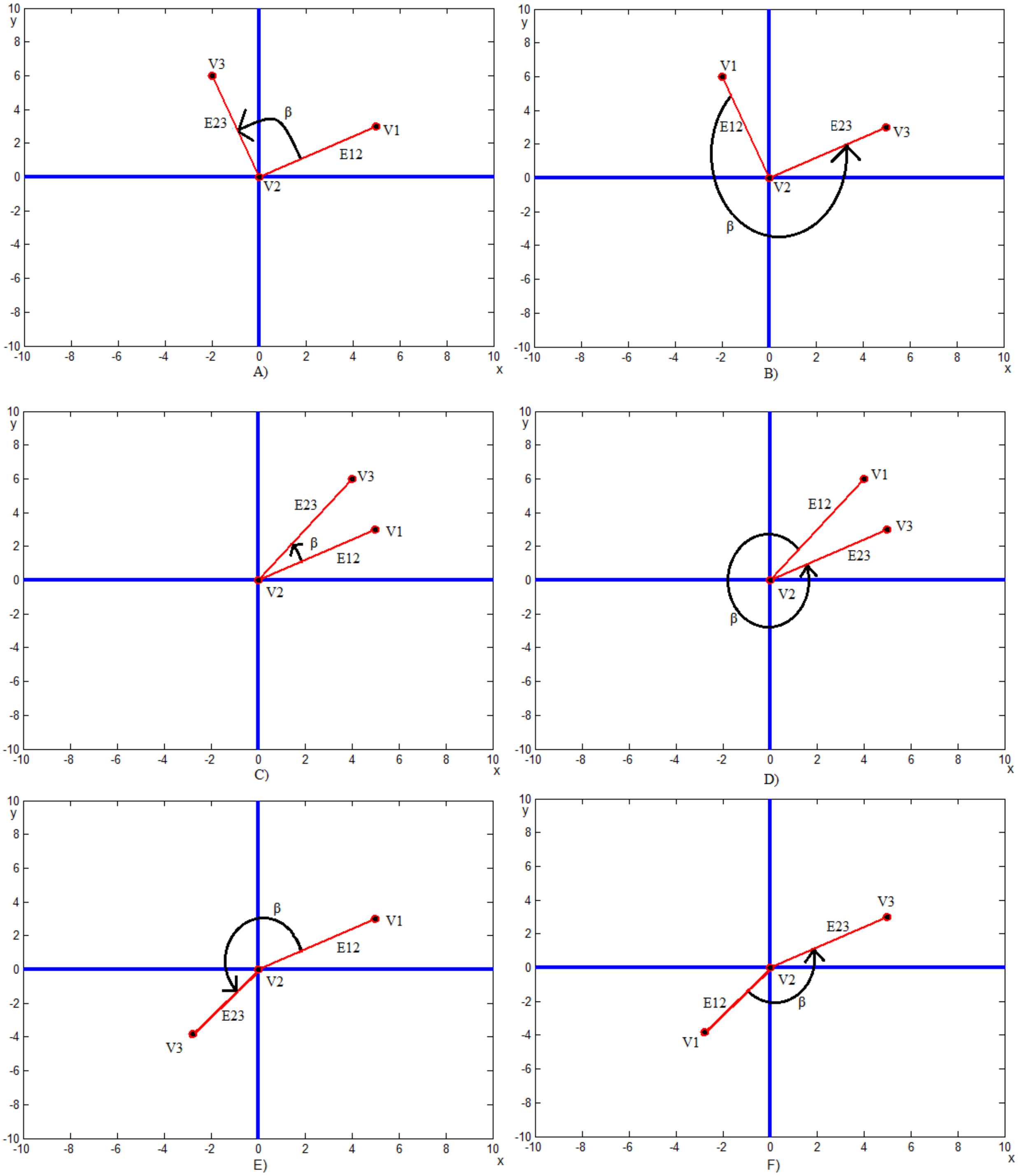

Figure 4.8 Angle beta in different situations

To find the inward and outward offsetting point V2N, a series of steps is proposed.

- Start with the initial vertex point found in the vertex table.
- Rename the vertex point to be evaluated to V2, the previous point as V1 and the point that follow V2 as V3. (Counterclockwise direction)
- Translate V1, V2, and V3 in the Cartesian plane so that V2 becomes the origin.
- Classify in which quadrant of the Cartesian plane V1 and V3 are found according to Figure 4.8.
- Calculate the value of β, which depend on the quadrant location and position of V1 and V3.
- Using the β value, find the location of V2N by trigonometry as the line V2-V2N is the angle bisector of the angle $\angle V1V2V3$.
- Translate V2N back in relation to V2's initial position.
- Find the parametric line parameters (T, U, V, and W) of the line V2-V2N.

  $T = (p - h); U = h; V = (q - k); W = k;$ where the x-y coordinates of points V2 and V2N are (p, h) and (q, k) respectively.
- Repeat the previous steps for all vertex points in the vertex table.
- Use the obtained parametric line equation to extrapolate the location of the inward offsetting vertex point for all vertex points in the vertex table.
- Using the parametric line equation, find the inward, and outward vertex points for any distance in relation with the α-cut of the sensor uncertainty membership function.

However, the proposed offsetting algorithm introduces a problem in some situations. This is illustrated in Figure 4.9 where the inward offsetting solution by the proposed algorithm produces a self-intersecting polygon (the black line polygon), which is an invalid solution for the fuzzy B-rep. Proper polygons are required for map representation. Therefore, a proper polygon test is devised. If the offset solution is self-intersecting, a modification algorithm is proposed to convert it to a proper polygon.

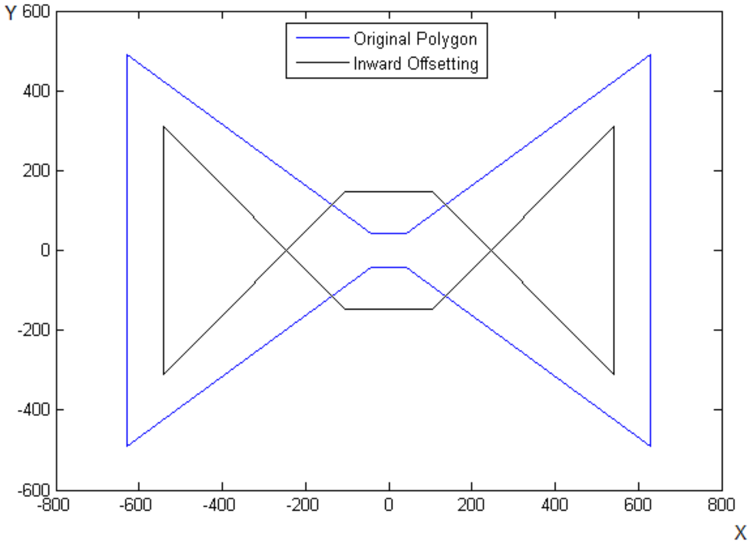

Figure 4.9 Inward offsetting solution of the blue line polygon

The self-intersection test checks for intersecting line segments in polygons because a proper polygon does not have any nonvertex intersection from all edges. Therefore, if at least one pair of the edges has nonvertex intersection in the resulting polygon, it is certain that the tested polygon is self-intersecting. Bentley [50] suggested an algorithm for detection of geometric intersections, which is very useful in detecting nonvertex intersections. This algorithm generates useful information for proper polygon conversion. Once the self-intersecting polygon test gives a positive result, an algorithm to convert the self-intersecting polygon to a proper polygon is carried out. In most situations, this conversion results in one polygon. However, applying the conversion algorithm to Figure 4.9 will result in two proper polygons. Nevertheless, if the conversion generates two proper polygons, one of them is treated as a hole and the other as the principal polygon (usually the polygon with more vertices and edges is chosen as principal and the other as a hole).

The conversion from a self-intersecting polygon to a proper polygon (if needed) marks the end of the sweeping operation.

4.2. Fuzzy Boundary Representation Data Structure

After the sweeping operation, all the information required to represent a fuzzy boundary map in a data structure is at hand. The vertex table and edge table represents the crisp B-rep data structure. A fuzzy B-rep map representation extends these two tables by adding necessary information like the sensor uncertainty membership function type, parameters, (if different membership functions are employed at different edges) and the parametric line equation parameters (T,U,V, and W) of the average normal vertex for every vertex in the vertex table.

A new table is added to store several indexed membership functions. This table is useful for describing which membership function is associated with edges of the polygon, and it provides the option of using different membership functions on different parts of the robot environment.

The crisp B-rep and fuzzy B-rep data structures are similar because the vertex and edge geometric information and their linking relation in the map remain the same. In the fuzzy B-rep data structure, additional columns are added in the vertex table including the line equation parameters of the average normal vertex points (VIN and VOUT) associated with the corresponding vertex point of the polygon and the membership functions associated with the corresponding edges. The Cartesian coordinates of the normal vertex points (VIN and VOUT) are also include in the fuzzy vertex table.

A α-cut of a fuzzy B-rep map consisting of the inward and outward offsets from the original polygon is shown in Figure 4.10. The resulting data structure is shown in Table 4.1, 4.2, and 4.3.

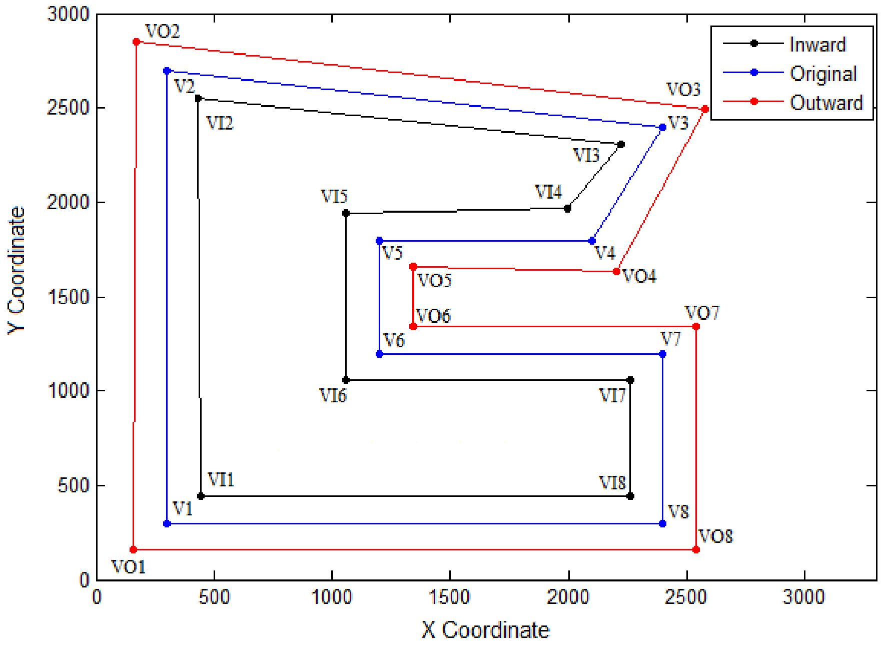

Figure 4.10 Fuzzy B-rep map example

V	**X**	**Y**	**VOUT**	**X**	**Y**	**VIN**	**X**	**Y**	**T**	**U**	**V**	**W**	**FM**
V1	300	300	**VO1**	158.58	158.58	**VI1**	441.42	441.42	-141.4	441.42	-141.42	441.42	1
V2	300	2700	**VO2**	168.96	2851.1	**VI2**	431.04	2548.9	-131.0	431.04	151.09	2548.9	1
V3	2400	2400	**VO3**	2577.2	2492.8	**VI3**	2222.8	2307.2	177.16	2222.8	92.82	2307.2	1
V4	2100	1800	**VO4**	2205.1	1629.9	**VI4**	1994.9	1970.1	105.15	1994.9	-170.1	1970.1	1
V5	1200	1800	**VO5**	1341.4	1658.6	**VI5**	1058.6	1941.4	141.42	1058.6	-141.42	1941.4	1
V6	1200	1200	**VO6**	1341.4	1341.4	**VI6**	1058.6	1058.6	141.42	1058.6	141.42	1058.6	1
V7	2400	1200	**VO7**	2541.4	1341.4	**VI7**	2258.6	1058.6	141.42	2258.6	141.42	1058.6	1
V8	2400	300	**VO8**	2541.4	158.58	**VI8**	2258.6	441.4	141.42	2258.6	-141.42	441.42	1

Table 4.1 Fuzzy Vertex Table

The fuzzy vertex table shown in Table 4.1 contains all the information of the crisp B-rep (the first three columns) and additional data. The table contains the coordinates of the outward vertex point (VOUT), the coordinates of the inward vertex point (VIN), the parametric line equation parameters (T, U, V, and W), and the FM that is an index to the sensor uncertainty membership function used in the corresponding vertex. This table follows the same indexing as the crisp B-rep vertex table. The crisp B-rep information and fuzzy B-rep information are maintained in separate columns. This allows convenient use of crisp B-rep information when performing the union operation of polygons for map building.

A new table is created (see Table 4.2), to store parameters of sensor uncertainty fuzzy membership functions associated with the map. Every used fuzzy membership function is indexed. The index works as a pointer to the assigned membership function for each vertex point, and the table contains its formula description and the support of the membership function.

Index	Parameters	Type
1	1. $a = -200$; 2. $b = +200$; 3. $TME = 200$;	Triangular 4. $\mu_x = \frac{(x-a)}{(-a)};\ when\ x \in [-TME, 0)$ 5. $\mu_x = \frac{(b-x)}{(b)};\ when\ x \in [0, +TME]$

Table 4.2 Membership function

For example, the triangular membership function is defined as $a = r - TME, b = r + TME, \mu_x = \frac{(x-a)}{(r-a)}, when\ x \in [r - TME, r)$ and $\mu_x = \frac{(b-x)}{(b-r)};\ when\ x \in [r, r + TME]$. Since the sensor uncertainty fuzzy set is used in the sweeping operation, the description is centered along the crisp B-rep map generated from range sensor data (**r** is zero). Using the normal line equation in the fuzzy vertex table, the sensor uncertainty fuzzy set and the vertex point are related. Only one membership function is used in the example shown in Figure 4.10. A fuzzy B-rep map can be associated with more than one membership function and its index is listed in the FM column of the fuzzy vertex table (Table 4.1).

The last table needed for a complete description of a fuzzy B-rep map is the fuzzy edge table shown in Table 4.3. This table contains the index to the members of an edge of the original polygon (crisp B-rep data), the inward and outward polygons.

Edge	Vstart	Vend	Edge OUT	Vstart	Vend	Edge IN	Vstart	Vend
E1	V1	V2	EO1	VO1	VO2	E1	VI1	VI2
E2	V2	V3	EO2	VO2	VO3	E2	VI2	VI3
E3	V3	V4	EO3	VO3	VO4	E3	VI3	VI4
E4	V4	V5	EO4	VO4	VO5	E4	VI4	VI5
E5	V5	V6	EO5	VO5	VO6	E5	VI5	VI6
E6	V6	V7	EO6	VO6	VO7	E6	VI6	VI7
E7	V7	V8	EO7	VO7	VO8	E7	VI7	VI8
E8	V8	V1	EO8	VO8	VO1	E8	VI8	VI1

Table 4.3 Fuzzy Edge Table

If the offset polygon is self-intersecting, the number of vertices of the polygon generated by the sweeping operation and proper polygon conversion may not be equal to that of the original polygon. In this case, the information for the deleted vertex points are represented as “NULL” to keep the same indexing for the rest of the table.

The proposed fuzzy B-rep map can be considered as an approximation to the fuzzification of the crisp B-rep map using the Extension Principle. At each membership grade α, all possible boundaries of the open area in the robot environment lie inside the α-cut of the fuzzy boundary set characterized by the generated fuzzy B-rep map. In particular, the support (α=0) of the fuzzy B-rep map provides an appropriate safety margin for robot navigation in the environment without collision at the highest confidence because its inner boundary bounds the space that the robot can navigate freely. On the other hand, the core (μ=1) of the fuzzy B-rep map is the same as the crisp B-rep map.

4.3. Summary

This chapter proposed algorithms for generation of all α-cuts of the fuzzy B-rep map of the robot environment by sweeping the sensor uncertainty fuzzy membership function along the boundary represented by the crisp B-rep map. Details of how the sensor uncertainties fuzzy membership function is constructed were presented. Sweeping operation is performed by offsetting the crisp B-rep polygon according to the required membership grade. A modified offsetting algorithm for the sweeping operation with the advantage of allowing efficient computation of α-cuts of the fuzzy set of the robot environment boundaries was proposed. The new data structure for the fuzzy B-rep map was presented. All necessary information is stored in three tables, namely fuzzy vertex table, fuzzy edge table, and sensor uncertainty membership functions table.

Chapter 5

Experimental results and discussions

This Chapter presents the experiment of building the Fuzzy Boundary Representation map of the fifth floor of the E1 EITC building using a Pioneer 3DX mobile robot.

5.1. Experimental Platform

A Pioneer 3DX mobile robot, which is equipped with a Laser range finder (LMS200), and an odometry sensor, is employed to test the proposed algorithm. To control the sensor measurements and motion of the robot, a MAC mini computer (A1176 Model) was used to run the control software for taking laser range scans and to drive the robot around. The measured range data is saved in a file and is sent through wireless transmission to a host laptop computer. A Matlab program runs on this laptop to read the files and execute the proposed algorithm. In Figure 5.1, the robot Pioneer 3DX with the Laser range finder and the MAC mini computer with the wireless transmission is shown.

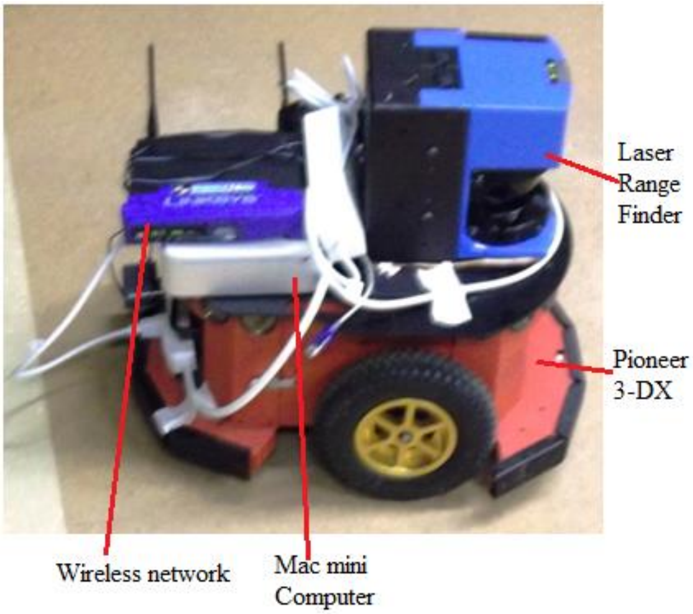

Figure 5.1 Experimental Platform

5.1.1. Robot Specifications

The following lists the technical specifications of the Pioneer 3DX robot.

Construction:

- Body: 1.6 mm aluminum (powder-coated)
- Tires: Foam-filled rubber

Operation:

- Robot Weight: 9 kg
- Operating Payload: 17 kg

Differential Drive Movement:

- Turn Radius: 0 cm
- Swing Radius: 26.7 cm

- Maximum Forward/Backward Speed: 1.2 m/s
- Rotation Speed: 300°/s
- Maximum Traversable Step: 2.5 cm
- Maximum Traversable Gap: 5 cm
- Maximum Traversable Grade: 25%
- Traversable Terrain: Indoor, wheelchair accessible

Power:

- Run Time: 8-10 hours with three batteries (with no accessories) Charge Time: 12 hours (standard) or 2.4 hrs. (Optional high-capacity charger)
- Available Power Supplies:

  5 V @ 1.5 A switched

  12 V @ 2.5 A switched

Additional Pioneer 3DX specification can be found in the datasheet provided online by adept mobiles robots [51].

5.1.2. Laser Range Finder Specifications

The following table lists the technical specifications of the LMS200 laser range finder installed on the Pioneer 3DX robot.

Scanning angle (field of vision)	180°
Motor speed	75HZ
Angular resolution (response time)	0.25° (53.33 ms); 0.5° (26.66 ms); 1° (13.33 ms); selectable
Range	Maximum 32M
Measurement resolution	10mm
Measurement accuracy	Typical ±35mm
Systematic error	– mm-mode: typical ±15 mm at range 1 to 8 m – cm-mode: typical ±4 cm at range 1 to 20 m
Statistical error	mm-mode: typical 5 mm at range ≤ 8 m/ reflectivity ≥ 10 %/ light ≤ 5 klx

Table 5.1 Laser Range finder technical specification

Additional information about the LMS200 manufactured by SICK Sensor Intelligence can be found in the datasheet [44].

The MAC mini has a 2.0 GHz Dual core CPU and 1GB Ram. It runs on Ubuntu 10 OS. A program developed by QT Library [52] is used to control the laser scan measurements. A power converter is used to power the Mac mini computer by converting the 12V DC power provided by onboard battery.

To verify the proposed algorithm, experiments were conducted to generate the map of the fifth floor of the E1 building of the Engineering and Information Technology Complex (EITC) of the University of Manitoba using fuzzy boundary representation. Figure 5.2 shows the floor plan of the mapping environment obtained from website of The University of Manitoba [53]. The environment is divided into four sections (A, B, C, D) that correspond to the four passages. Figures 5.3, 5.4, 5.5, and 5.6 shows the pictures four sections (A, B, C, and D respectively).

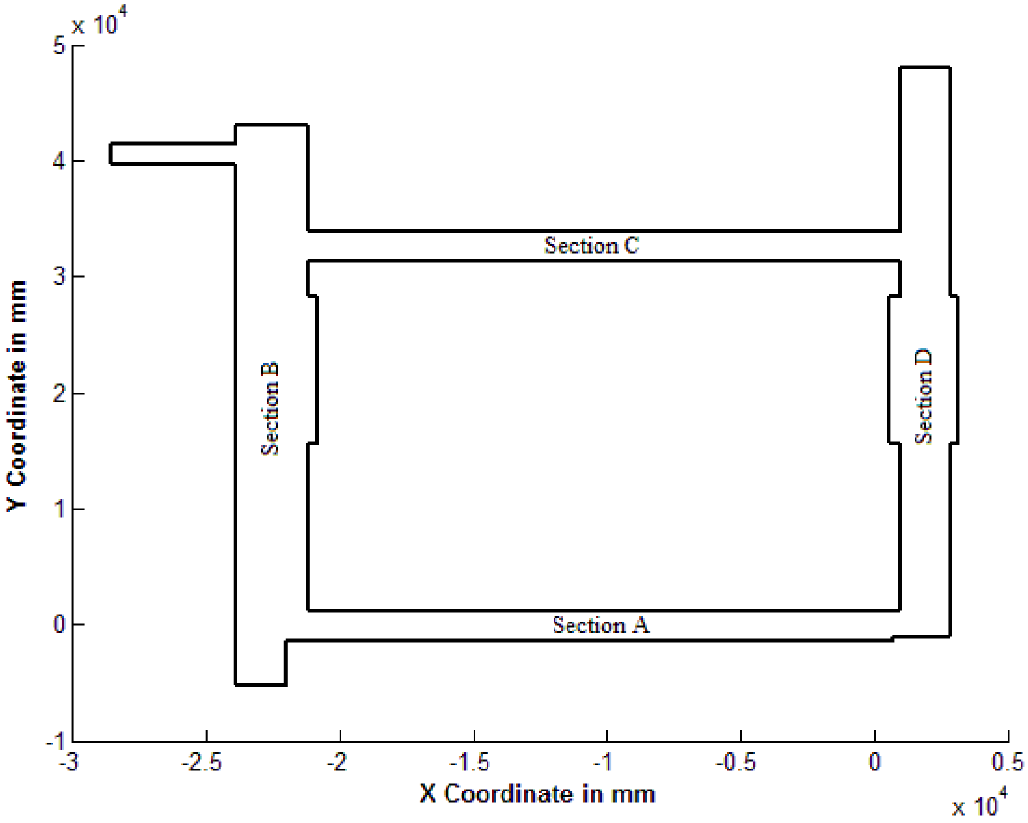

Figure 5.2 Map of the E1 fifth floor building

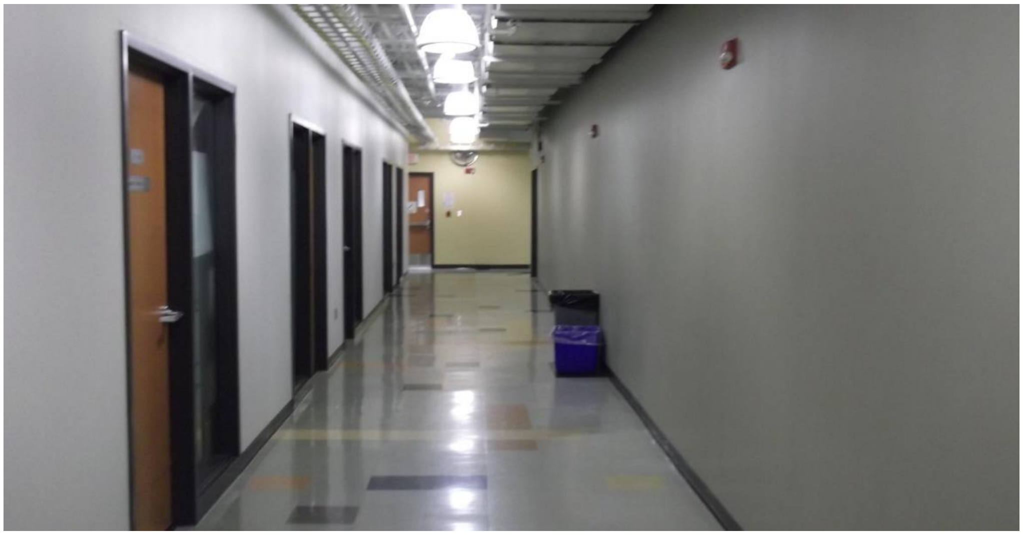

Figure 5.3 Section A

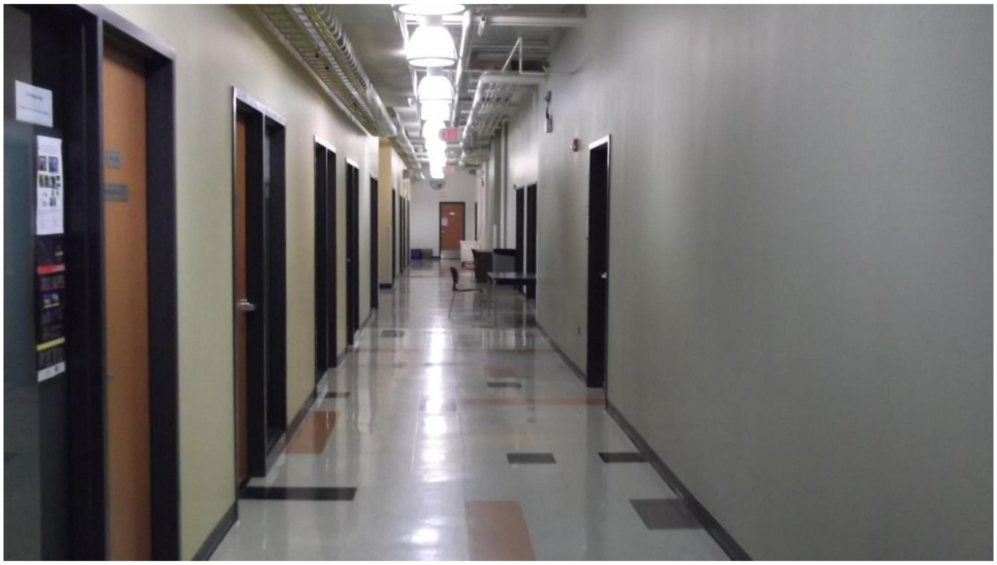

Figure 5.4 Section B

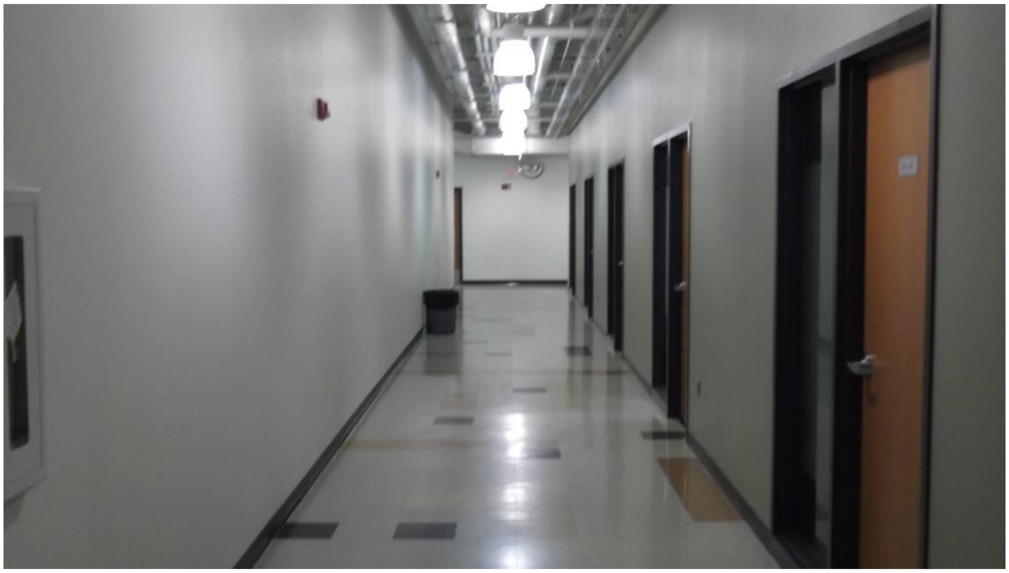

Figure 5.5 Section C

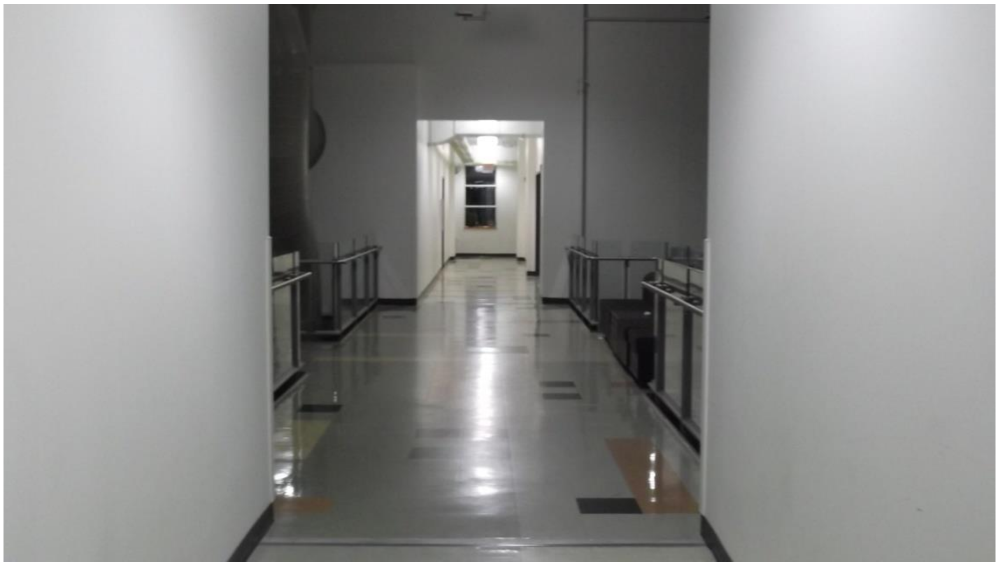

Figure 5.6 Section D

5.2. Experimental Procedure

The robot was programmed to follow a clockwise route from the Home position (blue circle) in Section A of the map as shown in Figure 5.7.

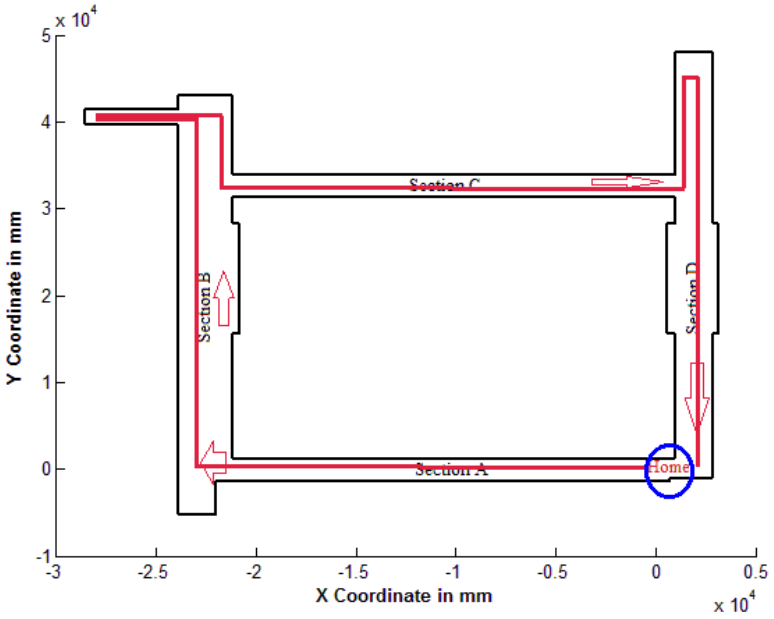

Figure 5.7 Robot route

A local scan was taken for every five meters. The laser range finder LMS200 takes a 180° measure. However, the data of each scan should cover 360°. This is achieved by consolidating two laser range measurements taken at local headings of 0° and 180°. The algorithm discussed in section 3.2.1 generated crisp polygons from the local range data. Local scan polygons were then combined to form the global crisp B-rep map of the environment by forming union of all local scan polygons. After that, the proposed algorithm discussed in Chapter 4 is applied to construct the fuzzy B-rep map. A time stamp of each local scan is useful to keep track of up to date information.

In the experiment, map building for section A, B and C, went well without any problems. However, excessive spikes appeared in local scans obtained in section D, as shown in Figure 5.7. This is due to the glass installed along the alley. Laser ray from the laser rangefinder, in fact, passed through the glass or was under specular reflection in section D. Figure 5.8 shows the consolidated polygon taken in section D.

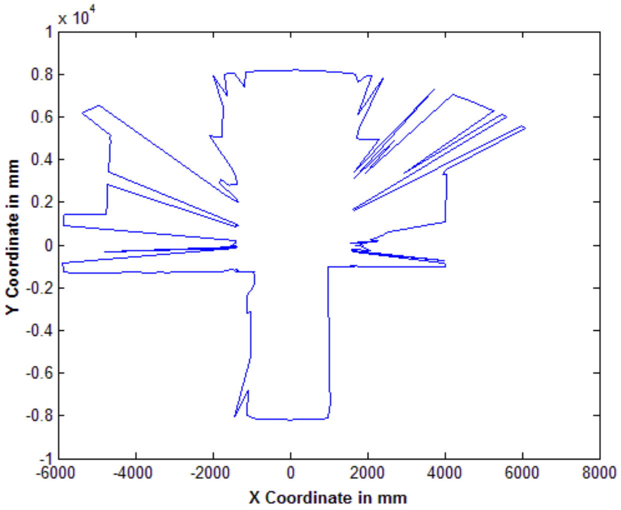

Figure 5.8 Noisy Section D data

The solution to this special situation was to use paper to cover the glass fence in section D as shown in Figure 5.9. Incorrect data was also taken in other locations. However, no modification was done to the environment unlike circumstance in section D. Differential filtering can be adopted in this kind of situations, but the filter configuration have to be adjusted to handle this specific situation. A Median filter is another solution for reducing spikes; however, because of the excessive spikes the output of the filter will also be highly inaccurate.

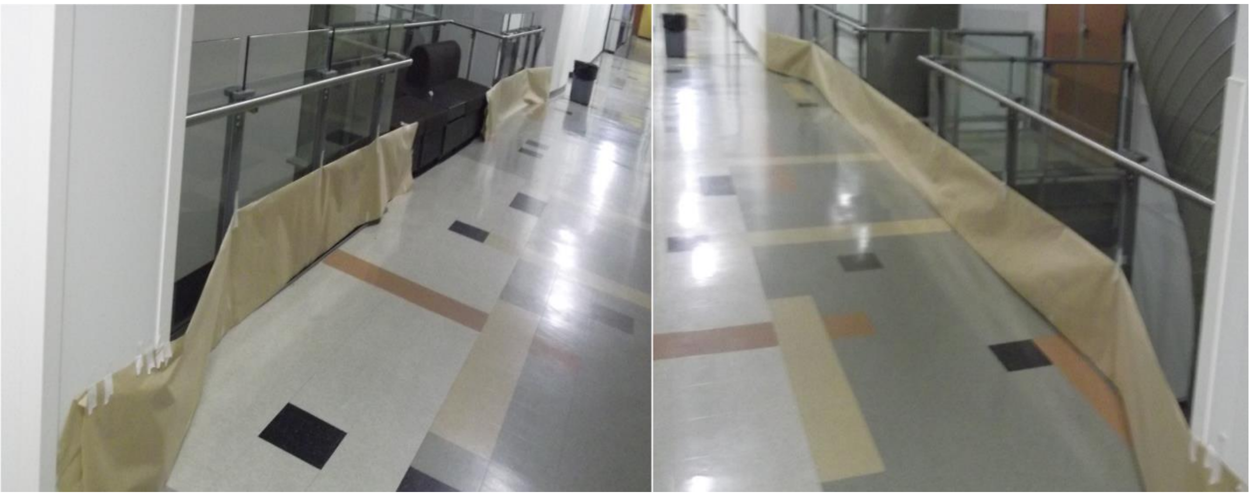

Figure 5.9 Section D with glass fence covered with paper

5.3. Results and Discussions

Before the generation of the fuzzy B-rep map, the membership function table for the sensor uncertainty fuzzy set has to be determined. As explained in Chapter 4, membership function can take different shapes and the triangular membership function is a reasonable choice for normal situations in which the robot does not have prior information of the environment. According to appendix I, the TME computed was 199.13mm. The following table shows the definition of the sensor uncertainty membership function employed in this experiment.

Index	Parameters	Type
1	6. $a = -199.13;$ 7. $b = +199.13;$ 8. $TME = 199.13;$	Triangular 9. $\mu_x = \frac{(x-a)}{(-a)};\ when\ x \in [-TME, 0)$ 10. $\mu_x = \frac{(b-x)}{(b)};\ when\ x \in [0, +TME]$

Table 5.2 Membership function Table

Once the membership function is described, the fuzzy B-rep map tables can be created according to algorithms proposed in Chapter 4. Figure 5.10 shows the support of the generated map (μ=0). The robot traveled approximately 140 meters in total, and 27 360° local laser range scans were processed.

The artifacts in section C were also produced by glass like in section D. However, no paper was used to cover the glass. Figure 5.11 shows the right area in section C where the artifact was presented (laser passed through the glass in range measurements). The artifact in section A was due to an open door.

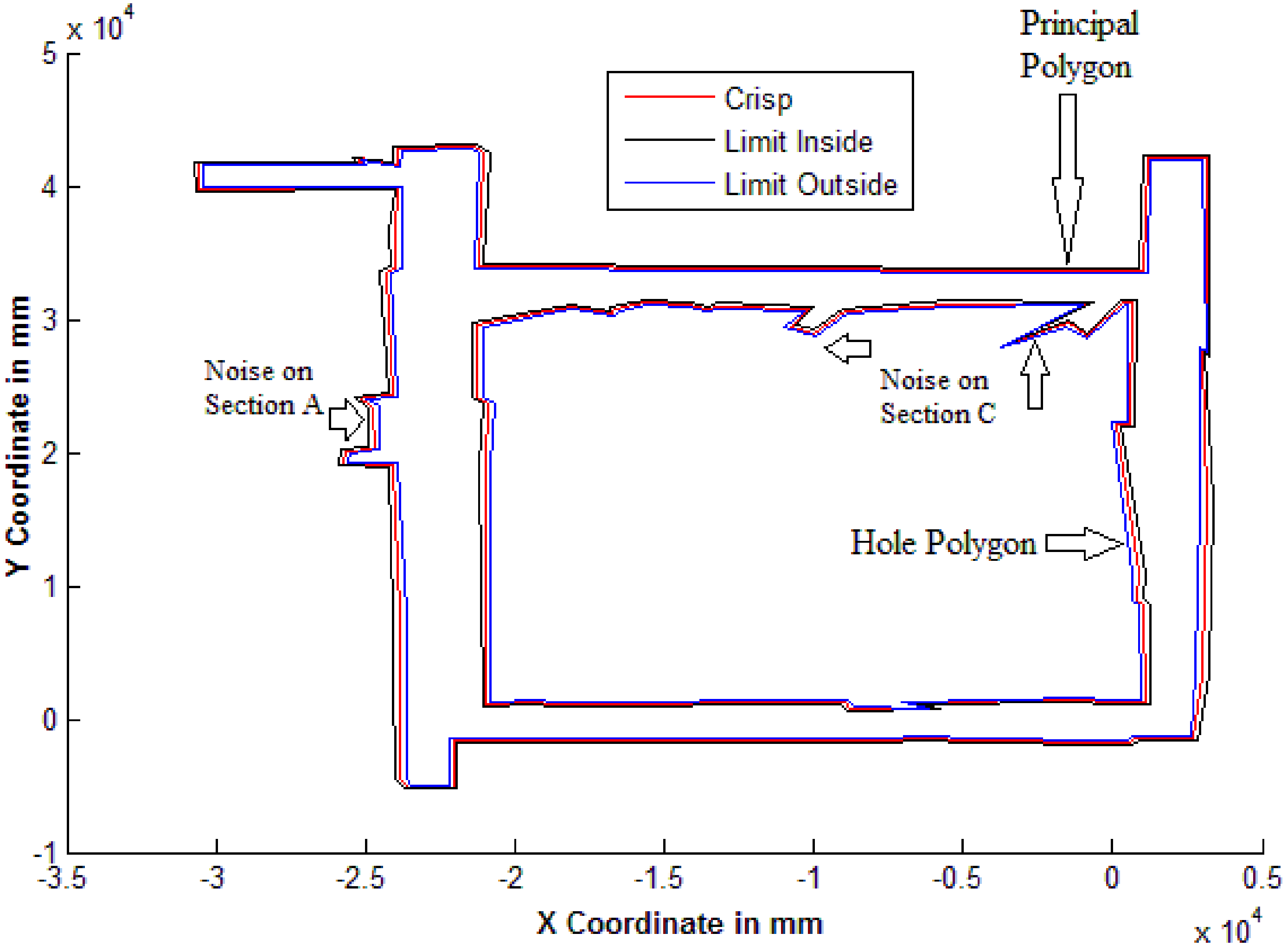

Figure 5.10 Support of the fuzzy B-rep map (μ=0)

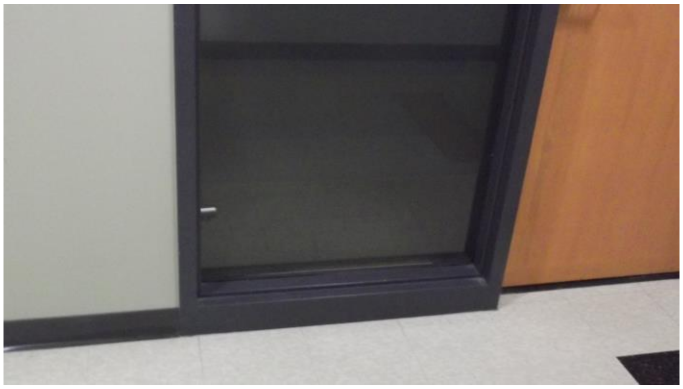

Figure 5.11 Glass boundaries on section C where the range data was noisy

Figure 5.12 shows a 3D plot of the membership function of the principal polygon. The figure shows the support and the core of the membership function (μ=0, μ=1 respectively). Figure 5.12 shows the value of the Cartesian coordinates of the vertex points (these vertex points represent the outer and inner boundary of the fuzzy map) when the membership grade is approximately μ=0.59.

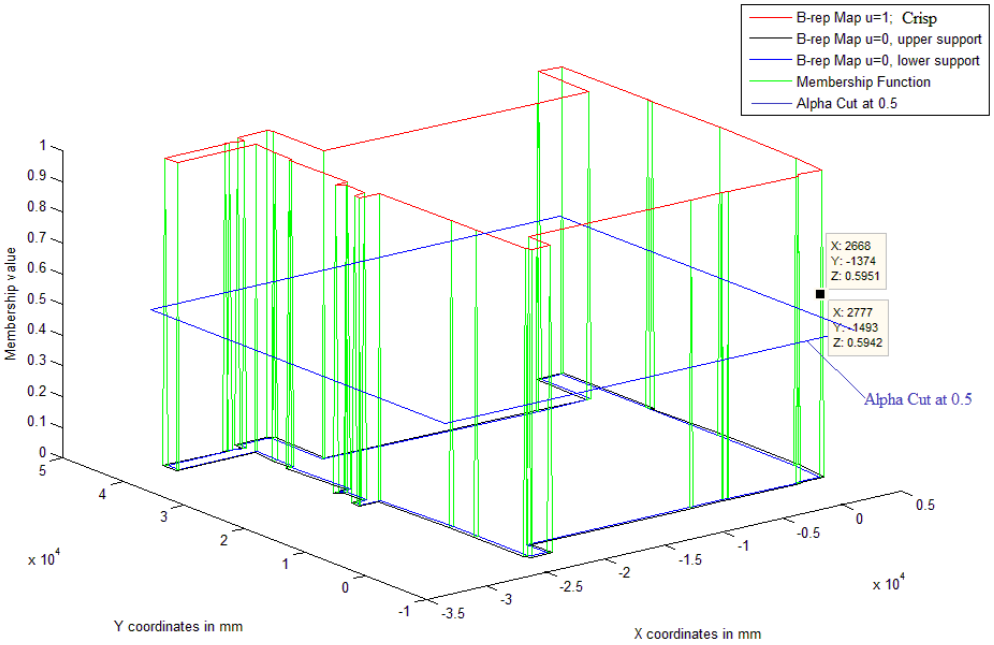

Figure 5.12 3D surface plot of the membership function for principal polygon

The resulting vertex table for the principal polygon has 40 vertices. The vertex table for the hole polygon has 37 vertices (both tables are listed in Appendix II). The edge tables of the principal and hole polygon are not presented in the thesis, because they only contain indices to the vertices of the corresponding edges. However, the edge table is part of the fuzzy B-rep map data structure and forms a data matrix of 40x6 for the principal polygon and 37x6 for the hole polygon.

The number of vertices of both principal and hole polygons can be further reduced by different methods. Least Mean Square Filter [54] and the Hough Transform [39] are algorithms that can be used for reducing the number of vertices. RANSAC [38] uses hypothetical models and test if the vertex points fitted that model (in this case, the model is a line). The algorithm chooses the model that fitted best over several points. On the other hand, Hough Transform tries to identify lines for several of points (See Chapter 3). Both methods have tuning parameters that affects the results. By using any of these methods, the complexity and memory requirements of the final map can be reduced in sacrifice of loss of details in the map.

Figure 5.13 shows the upper part of the section C of the fuzzy B-rep map with the floor plan. This figure shows that the floor plan boundary was bounded by the support of the fuzzy boundary. The inner bound of the fuzzy boundary support provides sufficient safety margin for autonomous navigation.

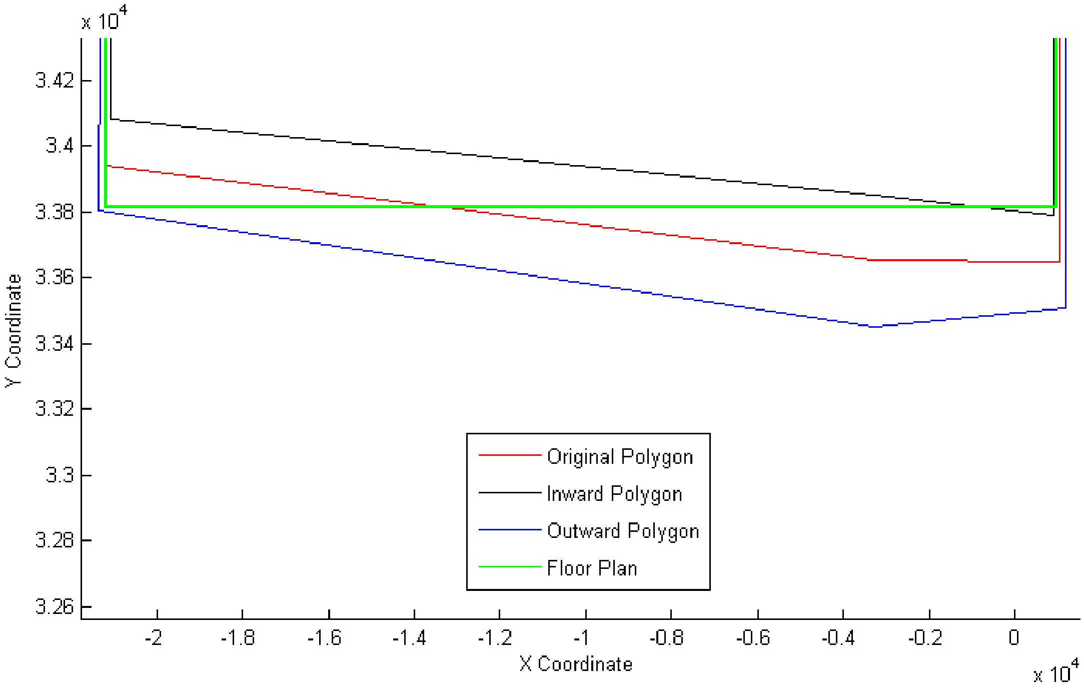

Figure 5.13 Comparisons between fuzzy map and floor plan (α=0)

The map shown in Figure 5.10 modeled an approximate area of 1050m². For instance, to store a map of this size using occupancy grid with a cell size of 30cmX30cm or 0.09 m² requires 11667 cells or floating-point numbers. On the other hand, the memory required for storage of the same environment in the fuzzy B-rep map are 1232 floating-point numbers (this includes the necessary information for the four tables discussed in Chapter 4), which correspond to 4928 bytes of memory.

As shown in Table 5.3, the memory space required for the fuzzy B-rep map does not depend on the physical dimension of the environment. On the other hand, the memory requirement for an occupancy grid map increases with the physical dimension of the environment and cells resolution. For instance, in Table 5.3 shows that changing the cell size from 30x30cm to 20x20cm doubles the used memory. Further decreasing the cell size (10x10 cm) quadruples the used memory. In comparison to fuzzy B-rep map data storage, an occupancy grid map with cell sizes of 30cmX30cm, 20cmX20cm, and 10cmX10cm requires 9, 21, and 85 times more memory respectively.

Fuzzy B-rep Map memory required In bytes	Probability Occupancy Grid Map (30x30cm cell size) In bytes	Probability Occupancy Grid Map (20x20 cm cell size) In bytes	Probability Occupancy Grid Map (10x10 cm cell size) In bytes
4928	46668	105000	420000

Table 5.3 Map storage size comparison for Figure 5.10

It is important to highlight that the proposed method tries to incorporate, but not to reduce sensor uncertainty information in robot map representation. Effective reduction of uncertainty in map representation usually relies on high computational complexity algorithms, which may not be able to implement on robot platform with limited computational and memory resources, especially in real-time. On the other hand, the sensor uncertainty information stored in fuzzy B-rep map provides reasonable tolerance in decision making or planning for robot tasks.

Chapter 6

Conclusions

6.1. Concluding Remarks

In robotics, sensors are the principal means of acquiring external environmental information. Ideally, information obtained by sensors is accurate. However, due to imperfect sensors, acquired sensor data is usually imprecise. The information obtained by sensors is used for the generation of an environmental map. A map is used for path planning, navigation and localization of the robot. Probabilistic methods are common for handling the uncertainty of the sensors. However, computational complexity and memory requirements limit the performance of these methods, like occupancy grid and Kalman Filter.

In this thesis, a way to incorporate sensors uncertainty into the robot environment map was investigated. The objective of the thesis is to include the sensors impreciseness into the boundary representation map structure with the use of fuzzy set theory. Fuzzy B-rep map is generated by converting the measured range data into scan polygons, then combining multiple scan polygons to form the resultant robot environment B-rep global map by performing the union operation, and finally fuzzification of the crisp B-rep map by sweeping the sensor uncertainty fuzzy set along the boundaries. The proposed method provides an approximation to fuzzification of boundaries of open area in a robot environment, instead of using the Extension principle, which is usually a computationally intensive task. This research demonstrated that the integration of the sensor uncertainty into boundary representation map structure is possible, and showed that the fuzzy B-rep data structure storage space is highly efficient in comparison to occupancy grid map structure. Experimental results verified that the support (μ=0) of boundaries of the fuzzy B-rep map of the fifth floor E1 of the EITC enclosed the floor plan boundary (See Chapter 5).

Not all the robot applications require precise environmental information. In real world, a simple task of moving to one location may be heavily taxed by the mapping method. Fuzzy B-rep map uses simple algorithms to convert range data into scan polygons, combining the scan polygons to generate a B-rep map, and fuzzification of the B-rep map to incorporate sensor uncertainty using the sweeping operation. Like humans, autonomous robots may not need precise information to complete navigation tasks.

6.2. Future works

The area of research presented in this thesis still has a wide range of aspects that need to be further studied. Accordingly, several future works are suggested:

1. The fuzzy B-rep map presented is in a 2D world and is straightforward to extend to 3D fuzzy B-rep. The use of B-rep data structure for the representation of solids (3D objects) is well known in solid modeling. However, the implementation of the sweeping operation to a 3D B-rep data structure using the sensor uncertainty membership function requires further developments.
2. Further research is required to handle dynamic environments. Dynamic environment refers to robot environment where the situation of robot surrounding area changes with time, like with moving obstacles, and so on. The use of time stamps for the measurements

can be useful for this kind of situations. New data takes priority over old data in map updates. Algorithm for union operation of time-stamped polygons will be developed.

3. Combination of different types of sensor information into map building. Gregory [55] proposed an algorithm called “just-in-time sensing” for combining Laser and Sonar range data by using the line segments intersection obtained by sonar and laser range data. Local scan polygons constructed from different types of sensors (laser range finder or ultrasonic sensors) can be combined by intersection operation to fuse information acquired by different sensor modalities in building robot environment maps using boundary representation.

Appendix I

Odometry uncertainties characterization

Odometry system also called the Dead-reckoning system is commonly used for robot positioning. Odometry sensors of a typical mobile robot are wheel encoders.

It is noteworthy that many researchers have developed algorithms that estimate the position uncertainty of a dead-reckoning robot [56] [57]. With this approach, each computed robot position is associated by a characteristic “error ellipse,” which shows a region of uncertainty for the robot's actual position (see Figure I.1) [56] [58]. While the robot traverses in an environment, the size of error ellipses continues to grow until a fixed position measurement is given to the system resetting the ellipse size. However, considering the systematic errors of odometry we may reduce the rate, which the error ellipses increase in relation to the distance traveled. Higher accuracy of positioning can be achieved without constantly using an external method to obtain a fixed position measurement. Nevertheless, this only slows the growing rate of the error ellipse. At time progresses the error ellipse continues to grow and will be in need of a reset. (It may be a position measurement using external system).

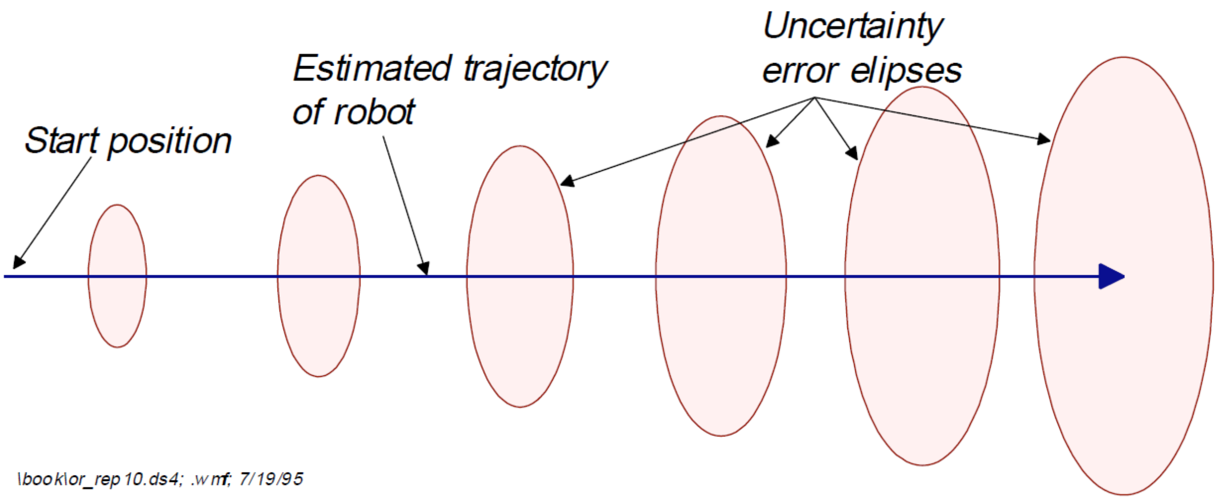

Figure I.1 Growing error ellipses indicates the growing uncertainty with odometry (Adapted from [56]**)**

The Pioneer 3-DX is equipped with a wheel encoder as an odometry system and it has a differential drive system. The odometry error can be classified into systematic errors and non-systematic errors.

Systematic errors are caused by:

- Unequal wheel diameters.
- Average of actual wheel diameters differs from the nominal wheel diameter.
- Actual wheelbase differs from nominal wheelbase.
- Misalignment of wheels.
- Encoder accuracy.

Non-systematic errors are caused by:

- Travel over uneven floors.
- Travel over unexpected objects on the floor.
- Wheel-slippage.

Estimating non-systematic errors is hard, but possible, if the robot environment is known apriori. Non – systematic errors may appear unexpectedly and can cause large position errors. However, if the robot has previous knowledge of the environment, it can detect non-systematic errors by continuously monitoring the position for large changes using finite difference methods [59]. On the other hand, systematic errors are always present in the odometry sensor measurements and can be easily estimated, and compensated [60].

I. Estimating Systematic errors of Pioneer 3DX robot

As mentioned in [61] [45] two types of the systematic errors are dominant and significant in robots with differential drive systems. They are the error due to unequal wheel diameters, defined as $E_d = {}^{D_R}/_{D_L}$ where D_R and D_L are the actual wheel diameters of right and left wheel respectively, and the error due to uncertainty about the effective wheelbase, defined as $E_b = {}^{b_{actual}}/_{b_{nominal}}$ where b is the wheelbase of the vehicle.

The systematic error due to difference between average of actual wheel diameters and nominal wheel diameter can usually be neglected. For example, a wheel encoder is set for a wheel diameter $D_{nominal}$ to travel $X_{nominal}$ distance for one revolution of the wheel. However, if the actual wheel diameter D_{actual} is different from the nominal one, the distance traveled in one revolution of the wheel X_{actual} will differ from $X_{nominal}$ and this difference introduces error in the system. Non-systematic error is insignificant in the environment where the mapping experiment was conducted in this thesis. This is because the main causes of the non-systematic error are not present in the experimental environment.

There are many ways of calculating the systematic error introduced by the odometry system including:

- *The Unidirectional Square-Path Test* [45]
- *The Bidirectional Square-Path Experiment or UMBmark* [45]
- *Calibration procedure to compensate for systematic error (by Goel et al* [62]*).*
- *An algorithm introduced by Roy and Thrun* [63].

The "UMBmark" is applied for obtaining the systematic error in the Pioneer 3-DX because it satisfies the need of a quantitative analysis of the odometry error. The procedure introduced by Goel et al., [62] requires a precise tachometer as reference and the Kalman filter to estimate the error. That complicates the calibration process. Roy and Thrun algorithm's [63] calibrates the odometry systematic error automatically while the robot operates using incremental maximum likelihood method with high computational complexity. The unidirectional square path test is not suitable for a differential-drive system like the Pioneer 3-DX robot because it can easily conceal two mutually compensating odometry errors.

II. The Bidirectional Square-Path Experiment or UMBmark

In this test, the robot is commanded to run in a 4mx4m square path with a fixed start position. End position is then read from an odometry sensor and is compared with the start position. The test is run 5 times in clockwise (cw) path and counterclockwise (ccw) path. The stopping positions of the cw and ccw paths are then clustered in two different areas. Next, the coordinates of the two centers of gravity of the clusters are computed using equation (1)

$$X_{c.g.,cw/ccw} = \frac{1}{n}\sum_{i=1}^{n} X_{i,cw/ccw}$$

Equation (1)

$$Y_{c.g.,cw/ccw} = \frac{1}{n}\sum_{i=1}^{n} Y_{i,cw/ccw}$$

where n is the number of run in each direction. Subsequently we continue to calculate the absolute offset for clockwise and counterclockwise paths given by equation (2)

$$r_{c.g,cw} = \sqrt{\left(X_{c.g.,cw}\right)^2 + \left(Y_{c.g.,cw}\right)^2}$$

Equation (2)

$$r_{c.g,ccw} = \sqrt{\left(X_{c.g.,ccw}\right)^2 + \left(Y_{c.g.,ccw}\right)^2}.$$

Finally, the larger value between $r_{c.g,cw}\ and\ r_{c.g,ccw}$ is considered the "measure of odometry accuracy for systematic errors":

$$E_{max,syst} = \max\left(r_{c.g,cw}, r_{c.g,ccw}\right).$$ Equation (3)

More detail information about the UMBmark test can be referred in [45].

III. Pioneer 3-DX UMBmark test results

The start position is considered as $P_{start} = (0,0)$ the obtained stopping position for cw and ccw paths in different runs are:

Number of run (n)	Stop position cw ($\boldsymbol{X_{n,cw}}$) and ($\boldsymbol{Y_{n,cw}}$) in mm	Stop position ccw ($\boldsymbol{X_{n,ccw}}$) and ($\boldsymbol{Y_{n,ccw}}$) in mm
1	(1327,645)	(-928,1477)
2	(1420,581)	(-893,1512)
3	(1318,594)	(-802,1225)
4	(1378,612)	(-879,1286)
5	(1349,642)	(-713,1289)

Table I.1 Stop Position Clockwise and Counter Clockwise before the calibration

Then the mean value defines the centers of gravity of the stop position clusters:

$$X_{c.g.,cw} = 1358.4 \quad Y_{c.g.,cw} = 614.8$$

$$X_{c.g.,ccw} = -843 \ \ Y_{c.g.,ccw} = 1357.8$$

And the absolute offset is given by equation (3):

$$r_{c.g.,cw} = 1491$$

$$r_{c.g.,ccw} = 1598.2$$

Finally, the systematic error is given by equation (4):

$$\left[E_{max,sys} = 1598.2\right]$$

In Figure I.2 we can see the result of the UMBmark test applied to the Pioneer 3-DX in a 4m² path.

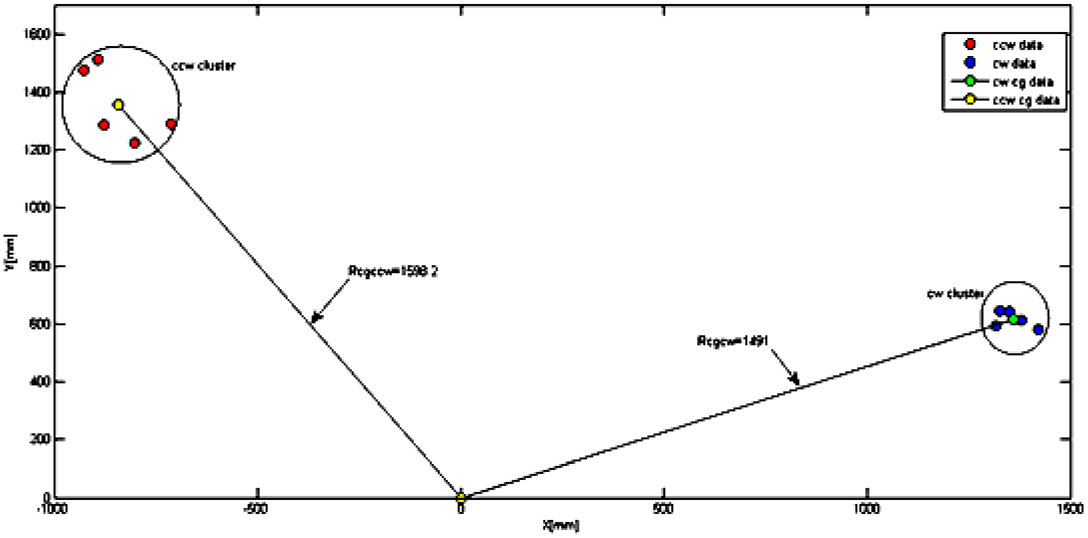

Figure I.2 Result from running UMBmark in 4m square path in the Pioneer 3-DX

IV. Correction of systematic errors for the Pioneer 3-DX

The Pioneer 3-DX has three calibration parameters. The first one is the "driftFactor" which is a value added or subtracted from the left-wheel encoder count at each motor cycle. It compensates for the effect of tire difference in the robot's forward and backward translation. This calibration parameter is closely related to a type B error in the systematic calibration introduced by Borenstein and Feng, L. [45] where this error is due to unequal wheel diameters, defined as $E_d = {}^{D_R}/_{D_L}$. The second calibration parameter is the "revCount" value, which is the differential number of encoder ticks for a 180-degree turn of the robot. It depends on several factors, principally the length of the wheelbase that may change due to payload, tire wear, operating surfaces and so on. This calibration parameter is related to the error introduced by E_b. $E_d and\ E_b$ are used as references for calculating the "driftFactor, and revCount." The third parameter "TicksMM" is unchanged.

Using the equations for the systematic calibration obtained in [45]:

$E_d = {}^{D_R}/_{D_L} = \frac{R+{}^{b}/_{2}}{R-{}^{b}/_{2}}$ Equation (4)

$E_b = \frac{b_{actual}}{b_{nominal}} = \frac{180°}{180°-\alpha}$ Equation (5)

$\alpha = \frac{X_{c.g.,cw}+X_{c.g.,ccw}}{-4L}\frac{180°}{\pi}$ Equation (6) (L is the length of one side of a square path)

$R = \frac{{}^{L}/_{2}}{\sin({}^{\beta}/_{2})}$ Equation (7)

$\beta = \frac{X_{c.g.,cw}-X_{c.g.,ccw}}{-4L}\frac{180°}{\pi}$ Equation (8)

Where $b = wheelbase = 267mm$ for the Pioneer 3DX robot.

Using equations (6) (8) we get:

$$\beta = 7.8832°$$

$$\alpha = -1.8456°$$

From equation (7):

$$R = 29095.3mm$$

and from equation (4):

$$E_d = 1.053 = {}^{D_R}/_{D_L}$$

Based on the result of E_d the right wheel diameter is a bit bigger than the left wheel diameter resulting in a curved traversal path when the robot is commanded to move a long straight line.

The “driftFactor” modifies the traveled distance reading in one revolution when the robot is commanded to move along a straight line. One revolution on the wheel corresponds to 195mm distance traveled (wheel size obtained from [46]). The “driftFactor,” (df), is computed as:

$$df = \big((195mm * E_d) - 195mm/rev\big) * 10 = 103.35mm$$

$$E_b = \frac{b_{actual}}{b_{nominal}} = \frac{180°}{180° - \alpha} = 0.99$$

The default “revCount” value, which was used for the UMBmark test, is 16570. Based on the computed E_b and applying the “Rule of Three” the revCount (revC) is computed as:

$$revC = \frac{16570}{E_b} = 16737.37$$

The driftFactor was changed to 104mm and the revCount to 16737, the test of the parameters gave minimum general error, but the clockwise and the counter clockwise error were unbalanced. The revCount value was then tuned to 16653 to balance the errors in clockwise and counterclockwise rotations.

After error compensation, the UMBmark test was repeated and stopping positions of the five runs are listed as follows.

CW	X	Y	CCW	X	Y
#1	-160	-150	#1	73	-75
#2	-134	-134	#2	102	-94
#3	-144	-140	#3	132	-114
#4	-117	-130	#4	136	-118
#5	-93	-100	#5	127	-87

Table I.2 Stop Position Clockwise and Counter Clockwise after the calibration

$$X_{c.g.,cw} = -129.6 \quad Y_{c.g.,cw} = -130.8$$

$$X_{c.g.,ccw} = 114 \quad Y_{c.g.,ccw} = -97.6$$

$$r_{c.g.,cw} = 184.13$$

$$r_{c.g.,ccw} = 150.07$$

The $r_{c.g.,cw}$ is then taken as the odometry sensor error as it is the largest error of both, clockwise and counter-clockwise runs, and it is used for the calculation of the Total Maximum Error (TME) in Chapter 4.

Appendix II

The Principal polygon and the hole polygon

The Vertex tables (II.1 and II.2) of the fuzzy B-rep map generated for the fifth floor of the E1 building of EITC are listed as follows.

VP Crisp		VP outside		VP Inside		Parametric Line Equation			
X	Y	XOUT	YOUT	XIN	YIN	T	U	V	W
2986.353	3315.948	3185.366	3309.125	2787.34	3322.77	-199	3185.366	6.8224	3309.13
3154.172	16171.72	3353.302	16171.72	2955.042	16171.72	-199.1	3353.302	0	16171.7
2991.055	27739.18	3034.548	27544.86	2947.562	27933.51	-43.49	3034.548	194.32	27544.9
3158.367	27370.76	3200.786	27176.2	3115.948	27565.32	-42.42	3200.786	194.56	27176.2
3113.39	42087.86	3255.815	42227.03	2970.965	41948.69	-142.4	3255.815	-139.2	42227
1149.044	42139.3	1011.035	42282.84	1287.053	41995.75	138.01	1011.035	-143.5	42282.8
1037.224	33647.74	897.388	33789.51	1177.06	33505.97	139.84	897.388	-141.8	33789.5
-3298.81	33650.18	-3298.81	33849.31	-3298.81	33451.05	0	-3298.81	-199.1	33849.3
-21196.8	33940.56	-21053.6	34078.91	-21340	33802.21	-143.2	-21053.6	-138.4	34078.9
-21040.9	42439.61	-20850.7	42498.62	-21231.1	42380.6	-190.2	-20850.7	-59.01	42498.6
-21375.2	42907.86	-21287.6	43086.66	-21462.9	42729.07	-87.67	-21287.6	-178.8	43086.7
-23896.1	42809.46	-24037.2	42949.97	-23755	42668.94	141.1	-24037.2	-140.5	42950
-23937.7	41617.82	-24052.6	41780.44	-23822.8	41455.2	114.93	-24052.6	-162.6	41780.4
-25311.4	42050.94	-25469.4	42172.1	-25153.3	41929.78	158.03	-25469.4	-121.2	42172.1
-25118.7	41749.08	-25293.8	41843.97	-24943.6	41654.18	175.07	-25293.8	-94.89	41844
-30581.2	41698.18	-30723.7	41837.24	-30438.7	41559.12	142.53	-30723.7	-139.1	41837.2
-30550.6	39699.22	-30689.1	39556.12	-30412.1	39842.31	138.48	-30689.1	143.1	39556.1
-23884.9	39815.87	-24026.7	39676.09	-23743	39955.66	141.82	-24026.7	139.79	39676.1
-23991.6	36472.35	-24190.8	36474.64	-23792.5	36470.06	199.12	-24190.8	-2.29	36474.6
-23968.4	33863.68	-24152.3	33939.87	-23784.4	33787.49	183.98	-24152.3	-76.19	33939.9
-24306.9	33531.08	-24491.5	33605.7	-24122.2	33456.47	184.62	-24491.5	-74.62	33605.7
-24067.3	24320.46	-24223.4	24444.09	-23911.2	24196.82	156.1	-24223.4	-123.6	24444.1
-25187.2	24087.55	-25370.6	24165.07	-25003.8	24010.03	183.42	-25370.6	-77.52	24165.1
-24718.8	23350.29	-24910	23294.55	-24527.6	23406.03	191.17	-24910	55.739	23294.5
-24714.6	20330.32	-24872.7	20451.44	-24556.6	20209.21	158.06	-24872.7	-121.1	20451.4
-25656.1	20078.06	-25810.6	20203.64	-25501.6	19952.48	154.54	-25810.6	-125.6	20203.6
-25707.8	19151.2	-25857.3	19019.56	-25558.4	19282.85	149.41	-25857.3	131.64	19019.6
-24099.3	19037.61	-24243.4	18900.2	-23955.1	19175.02	144.12	-24243.4	137.41	18900.2
-23839.1	7637.338	-24038.3	7637.338	-23640	7637.338	199.13	-24038.3	0	7637.34
-23855.3	3572.427	-24054.4	3572.427	-23656.2	3572.427	199.13	-24054.4	0	3572.43
-23829.9	-4518.9	-24024.6	-4560.61	-23635.2	-4477.2	194.71	-24024.6	41.707	-4560.6
-23577	-5086.94	-23685.5	-5253.95	-23468.5	-4919.94	108.45	-23685.5	167	-5253.9
-22069.7	-5086.94	-21928.9	-5227.75	-22210.5	-4946.14	-140.8	-21928.9	140.81	-5227.7
-22069.7	-1583.16	-21929.1	-1724.12	-22210.4	-1442.19	-140.6	-21929.1	140.97	-1724.1
-8363.85	-1614.93	-8359.69	-1814.01	-8368	-1415.84	-4.158	-8359.69	199.09	-1814
-5676.89	-1496.4	-5714.43	-1691.96	-5639.34	-1300.85	37.544	-5714.43	195.56	-1692
-5298.51	-1666.94	-5341.24	-1861.43	-5255.79	-1472.44	42.724	-5341.24	194.49	-1861.4
565.0477	-1719.91	664.5863	-1892.37	465.5092	-1547.44	-99.54	664.5863	172.47	-1892.4
756.0897	-1382.24	854.146	-1555.56	658.0334	-1208.93	-98.06	854.146	173.31	-1555.6
2722.561	-1433.74	2857.503	-1580.17	2587.62	-1287.3	-134.9	2857.503	146.44	-1580.2

Table II.1 Fuzzy Vertex Table for principal polygon

VP Crisp		VP outside		VP Inside		Parametric Line Equation			
X	Y	XOUT	YOUT	XIN	YIN	T	U	V	W
151.69	22112.25	296.43	21974.22	6.9612	22250.28	-144.7	296.428	138.03	21974.2
880.18	10896.19	1080.2	10899.07	680.2	10893.31	-200	1080.16	-2.878	10899.1
810.94	8978	1005.9	9022.436	615.94	8933.564	-195	1005.946	-44.44	9022.44
1035.1	8551.828	1228.9	8601.085	841.25	8502.571	-193.8	1228.924	-49.26	8601.09
1133.1	1293.519	1275.2	1152.77	991.05	1434.268	-142.1	1275.229	140.75	1152.77
-450.6	1299.891	-487.5	1103.321	-413.7	1496.462	36.88	-487.47	196.57	1103.32
-815.8	1440.222	-848.4	1242.908	-783.1	1637.535	32.672	-848.426	197.31	1242.91
-6852	1206.467	-6657	1162.597	-7047	1250.337	-195.1	-6656.98	43.87	1162.6
-5942	731.5522	-5749	680.6505	-6136	782.4538	-193.4	-5748.69	50.902	680.651
-8772	826.9028	-8880	658.4741	-8665	995.3314	107.85	-8880.26	168.43	658.474
-8978	1236.445	-9082	1065.519	-8874	1407.37	103.85	-9081.92	170.93	1065.52
-18857	1103.055	-18903	908.3038	-18812	1297.806	45.519	-18902.9	194.75	908.304
-19221	1289.072	-19258	1092.476	-19184	1485.668	36.74	-19257.9	196.6	1092.48
-20951	1109.884	-21085	961.2011	-20817	1258.568	133.77	-21084.9	148.68	961.201
-20993	19098.68	-21193	19101.13	-20793	19096.24	199.99	-21192.9	-2.447	19101.1
-20871	23643.49	-21061	23579.67	-20682	23707.32	189.54	-21060.7	63.826	23579.7
-21265	24133.82	-21454	24067.81	-21076	24199.83	188.79	-21453.5	66.014	24067.8
-21245	29558.6	-21412	29668.56	-21078	29448.64	167.06	-21412	-110	29668.6
-18069	30939.68	-18072	31139.66	-18065	30739.71	3.2702	-18071.9	-200	31139.7
-16747	30415.52	-16832	30596.14	-16661	30234.91	85.903	-16832.4	-180.6	30596.1
-16640	30752.32	-16790	30885.38	-16491	30619.25	149.31	-16789.7	-133.1	30885.4
-15593	31220.63	-15629	31417.37	-15557	31023.9	35.991	-15628.6	-196.7	31417.4
-14226	31140.66	-14174	31333.85	-14278	30947.46	-51.73	-14174.5	-193.2	31333.8
-13544	30798.79	-13544	30998.79	-13544	30598.79	0	-13544.5	-200	30998.8
-13140	30992.98	-13177	31189.57	-13104	30796.39	36.775	-13177.1	-196.6	31189.6
-10169	30760.41	-9996	30860.87	-10342	30659.95	-172.9	-9996.03	-100.5	30860.9
-10775	29473.15	-10583	29527.47	-10968	29418.82	-192.5	-10582.9	-54.33	29527.5
-9997	28962.7	-10040	29157.92	-9953	28767.48	43.48	-10040.4	-195.2	29157.9
-8948	30664.88	-9058	30832.29	-8839	30497.46	109.42	-9057.83	-167.4	30832.3
-5386	31163.41	-5400	31362.91	-5372	30963.92	14.159	-5400.23	-199.5	31362.9
-811.6	31175.63	-629	31257.22	-994.2	31094.04	-182.6	-628.972	-81.59	31257.2
-3620	28056.13	-3473	28191.26	-3768	27920.99	-147.4	-3472.95	-135.1	28191.3
-1492	29661.13	-1461	29858.72	-1523	29463.54	-30.94	-1460.69	-197.6	29858.7
-860.8	28766.45	-877	28965.79	-844.6	28567.11	16.213	-877.04	-199.3	28965.8
343.63	31248.4	236.91	31417.55	450.34	31079.25	106.71	236.9118	-169.2	31417.5
645.64	31250.4	786.88	31392.01	504.4	31108.8	-141.2	786.8775	-141.6	31392
608.59	22104.28	748.48	21961.34	468.7	22247.21	-139.9	748.4836	142.93	21961.3

Table II.2 Fuzzy Vertex Table for hole polygon

Bibliography

[1] W. H. Huang and K. R. Beevers, "Topological mapping with sensing-limited robots," in *Algorithmic Foundations of Robotics VI*, New York, New York: Springer, 2004, pp. 367-382.

[2] A. Bücken and S. Thrun, "Integrating Grid-Based and Topological Maps for Mobile Robot Navigation," *in Proceeding of The Thirteenth National Conference on Artificial Intelligence (AAAI-96),* vol. 2, pp. 944-950, 1996.

[3] W. Burgard, S. Thrun and D. Fox, Probabilistic Robotics, Cambridge, Massachusetts: The MIT Press, 2005.

[4] A. Martín and J. Gasós, "Mobile Robot Localization Using Fuzzy Maps," in *Fuzzy Logic in Artificial Intelligence Towards Intelligent Systems*, vol. 1188, Berlin, Springer, 1997, pp. 207-224.

[5] R. Vázquez-Martín, P. Núñez, A. Bandera and F. Sandoval, "Curvature-Based Environment Description for Robot Navigation Using Laser Range Sensors," *Sensors,* vol. 9, no. 8, pp. 5894-5918, 2009.

[6] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering,* vol. 82, no. 1, pp. 35-45, March 1960.

[7] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," University of North Carolina at Chapel Hill, Chapel Hill, 1995.

[8] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE,* vol. 92, no. 3, pp. 401- 422, March 2004.

[9] S. A. Holmes, G. Klein and D. W. Murray, "An O(N^2) Square Root Unscented Kalman for Visual Simultaneous Localization and Mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 31, no. 7, pp. 1251 - 1263, 2009.

[10] L. Zadeh, "Fuzzy Sets," *Information and Control,* vol. 8, no. 3, pp. 338-353, 1965.

[11] G. J. Klir and B. Yuan, Fuzzy Sets and Fuzzy Logic, Upper Saddle River, NJ: Prentice Hall PTR, 1995.

[12] K. Warwick, A. Ekwue and R. Aggarwal, Artficial Intelligence Techniques in Power Systems, vol. 22, Herts: The Institution of Electrical Engineers, 1997.

[13] C. T. Lin and C. S. George Lee, Neural fuzzy systems : a neuro-fuzzy synergism to intelligent systems, Upper Saddle River, New Jersey: Prentice-Hall, 1996.

[14] W. Diedrich, Spatial Representation and Reasoning for Robot Mapping, vol. 48, Bremen: Springer, 2008.

[15] M. E. Jefferies and W.-K. Yeap, Robotics and Cognitive Approaches to Spatial Mapping, vol. 38, Berlin: Springer, 2008.

[16] K. Cheng, "A purely geometric module in the rat's spatial representation," *Cognition,* vol. 23, no. 2, pp. 149-178, 1986.

[17] L. Hermer, "Internally coherent spatial memories in a mammal," *Neuroreport,* vol. 8, no. 7, pp. 1743-1747, 1997.

[18] M. O. Franz, B. Schölkopf, P. Georg, H. A. Mallot and H. H. Bülthoff, "Learning View Graphs for Robot Navigation," *Autonomous Robots,* vol. 5, no. 1, pp. 111-125, 1997.

[19] M. Denis, "The description of routes: A cognitive approach to the production of spatial discourse," *Cahiers de psychologie cognitive,* vol. 16, no. 4, pp. 409-458, 1997.

[20] Elibol, Armagan, Gracias, Nuno, Garcia and Rafael, Efficient Topology Estimation for Large Scale Optical Mapping, vol. 82, Springer, 2013.

[21] G. Welch and G. Bishop, An Introduction to the Kalman Filter, NC: Chapel Hill, 2006.

[22] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer,* vol. 22, no. 6, pp. 46-57, 1989.

[23] M. L. Gavrilova, Generalized Voronoi Diagram: A Geometry-Based Approach to Computational Intelligence, Banff: Springer Publishing Company, Incorporated, 2010, pp. 3-8.

[24] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence,* vol. 99, no. 1, pp. 21-71, 1998.

[25] S. Thrun, "Robotic mapping: a survey," in *Exploring artificial intelligence in the new millennium*, San Francisco, CA: Morgan Kaufmann Publishers Inc., 2003, pp. 1-35.

[26] O. Punska, *Bayesian Approaches to Multi-Sensor Data Fusion,* M.Phil. thesis, University of Cambridge, Cambridge, United Kingdom, 1999.

[27] Y. Y. Yao, "A Comparative Study of Fuzzy Sets and Rough Sets," *Information Sciences,* vol. 109, no. 1-4, pp. 227-242, 1998.

[28] Y. Y. Yao, "Two views of the theory of rough sets in finite universes," *International Journal of Approximate Reasoning,* vol. 15, no. 4, pp. 291-317, 1996.

[29] Z. Pawlak, "Rough Sets," *International Journal of Computer and Information Sciences,* vol. 11, no. 5, pp. 341-356, 1982.

[30] M. K. Chakraborty, "On fuzzy sets and rough sets from the perspective of indiscernibility," *in Proceedings of the 4th Indian conference on Logic and its applications (ICLA '11),* pp. 22-37, 2011.

[31] G. Shafer, "Perspectives on the theory and practice of belief functions," *International Journal of Approximate Reasoning,* vol. 4, no. 5-6, pp. 323-362, 1990.

[32] T. D. Astride Aregui, "Constructing consonant belief functions from sample data using confidence sets of pignistic probabilities," *International Journal of Approximate Reasoning,* vol. 49, no. 3, pp. 575-594, 2008.

[33] T. D. Astride Aregui, "Constructing Predictive Belief Functions from Continuous Sample Data Using Confidence Bands," *in Proceeding of the 5th International Symposium on Imprecise Probability: Theories and Applications (ISIPTA '07),* vol. 1, pp. 11-20, 2007.

[34] A. G. Requicha, "Representations for Rigid Solids: Theory, Methods, and Systems," *ACM Computing Surveys (CSUR),* vol. 12, no. 4, pp. 437-464, 1980.

[35] S. Ghali, Introduction to Geometric Computing, Springer, 2008.

[36] H. Christoph M., Geometric & Solid Modeling: An Introduction, San Mateo, California: Morgan Kaufmann, 1989, pp. 36-46.

[37] M. A. Fischler and R. C. Bolles, Readings in computer vision: issues, problems, principles, and paradigms, San Francisco, CA: Morgan Kaufmann Publishers Inc., 1987.

[38] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM,* vol. 24, no. 6, pp. 381-395, 1981.

[39] R. Duda and P. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM,* vol. 15, no. 1, pp. 11-15, 1972.

[40] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Boston, MA, USA: Addison-Wesley Longman Publishing Co., 1992.

[41] A. K. Jain, Fundamentals of digital image processing, Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[42] D. Ballard and C. Brown, Computer Vision, Prentice-Hall, 1982.

[43] W. R. Franklin, "PNPOLY - Point Inclusion in Polygon Test," 31 December 2009. [Online]. Available: http://www.ecse.rpi.edu/Homepages/wrf/Research/Short_Notes/pnpoly.html. [Accessed May 2012].

[44] Sick Sensor Intelligence, "LMS200 Laser Measurement System Technical Description," [Online]. Available: http://sicktoolbox.sourceforge.net/docs/sick-lms-technical-description.pdf. [Accessed May 2012].

[45] J. Borenstein and L. Feng, "UMBmark: A Benchmark Test for Measuring Deadreckoning Errors in Mobile Robots," *in Proceeding of the 1995 SPIE Conference on Mobile Robots (SPIE'95),* vol. 2591, pp. 113-124, 1995.

[46] R. Wein, "Exact and approximate construction of offset polygons," *Computer-Aided Design,* vol. 39, no. 6, pp. 518-527, 2007.

[47] P. K. Agarwal, E. Flato and D. Halperin, "Polygon decomposition for efficient construction of Minkowski sums," *Computational Geometry: Theory and Applications - Special issue on: Sixteenth European Workshop on Computational Geometry (EUROCG '00),* vol. 21, no. 1-2, pp. 39-61, 2002.

[48] K. Su-Jin, L. Dong-Yoon and Y. Min-Yang, "Offset Triangular Mesh Using the Multiple Normal Vectors of a Vertex," *Computer-Aided Design and Applications,* vol. 1, no. 1-4, pp. 285-292, 2004.

[49] B. Koc and Y. S. Lee, "Non-uniform offsetting and hollowing objects by using biarcs fitting for rapid prototyping process," *Computers in Industry,* vol. 47, no. 1, pp. 1-23, 2002.

[50] J. L. Bentley and T. A. Ottmann, "Algorithms for reporting and counting geometric intersections," *IEEE Transactions on Computers,* vol. 28, no. 9, pp. 643-647, 1979.

[51] Adept Technology, Inc., "Pioneer 3-DX Datasheet," [Online]. Available: http://www.mobilerobots.com/Libraries/Downloads/Pioneer3DX-P3DX-RevA.sflb.ashx. [Accessed January 2012].

[52] A. Ezust and P. Ezust, Introduction to Design Patterns in C++ with Qt, Second ed., Upper Saddle River, NJ, USA: Prentice Hall Press, 2011.

[53] University of Manitoba, "University of Manitoba Campus Floor plans," [Online]. Available: http://umanitoba.ca/campus/physical_plant/fortgarry/pdfs/231a.pdf. [Accessed September 2012].

[54] B. K. Paul Horn, Robot Vision, Massachusetts, USA: MIT Press ed, 1986.

[55] G. Dudek, P. Freedman and I. M. Rekleitis, "Just-in-time sensing: efficiently combining sonar and laser range data for exploring unknown worlds," *in Proceeding of the International Conference on Robotics and Automation (ICRA '96),* vol. 1, pp. 667-671, 1996.

[56] Y. Tonouchi, T. Tsubouchi and S. Arimoto, "Fusion of Dead-reckoning Positions with a Workspace Model for a Mobile Robot by Bayesian Inference," *in Proceeding of the International Conference on Intelligent Robots and Systems (IROS '94),* vol. 2, pp. 1347-1354, 1994.

[57] K. Komoriya and E. Oyama, "Position estimation of a mobile robot using optical fiber gyroscope (OFG)," *in Proceeding of the International Conference on Intelligent Robots and Systems (IROS '94),* vol. 1, pp. 143-149, 1994.

[58] M. Adams, et al. "Control and Localization of a Post Distributing Mobile Robot," *in Proceeding of the International Conference on Intelligent Robots and Systems (IROS '94),* vol. 1, pp. 150-156, 1994.

[59] K. K. Autar, E. K. Egwu and N. Duc, Numerical Methods with Applications, 2nd. ed., Florida, USA: University of Florida, 2011.

[60] J. Borenstein, H. R. Everett and L. Feng, "Where I am? Sensors and Methods for mobile Robots Positioning," The University of Michigan, Michigan, 1996.

[61] J. Borenstein and L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots," *IEEE Transactions on Robotics and Automation,* vol. 12, no. 6, pp. 869-880, 1996.

[62] P. Goel, S. I. Roumeliotis and G. S. Sukhatme, "Robust localization using relative and absolute position estimates," *in Proceeding of the International Conference on Intelligent Robots and Systems (IROS '99),* vol. 2, pp. 1134-1140, 1999.

[63] N. Roy and S. Thrun, "Online self-calibration for mobile robots," *IEEE International Conference on Robotics and Automation,* vol. 3, pp. 2292-2297, 1999.