ROBOTIC MAPPING USING SOFT COMPUTING METHODOLOGIES

MOMOTAZ BEGUM

ROBOTIC MAPPING USING SOFT COMPUTING METHODOLOGIES

by

© Momotaz Begum, B.Sc (Eng).

A thesis Submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of

Master of engineering

C-CORE/ Faculty of Engineering and Applied Science
Memorial University of Newfoundland
July, 2005
St.John's, Newfoundland, Canada

Abstract

Robotic mapping has been an active research area in robotics and Artificial Intelligence (AI) in the last two decades. The research on robotic mapping focuses to solve the self-localization problem of a mobile robot while it is navigating through an unknown environment (either indoor or outdoor) in order to build a map of that environment from a series of sensor measurements collected by the robot itself. This joint problem of mapping and self localization is commonly referred to as Concurrent Mapping and Localization (CML) or Simultaneous Localization and Mapping (SLAM) in mobile robotics research. The existing techniques used to solve CML (or SLAM) include Kalman Filter (KF), Extended Kalman Filter (EKF), Expectation Maximization (EM) algorithm, Particle Filter and different combinations thereof. The objective of this research is to develop a novel robotic mapping algorithm for indoor and outdoor environments using soft computing methods.

The proposed algorithm formalizes the robotic mapping process as an optimization problem. The objective function measures the fitness of a robot pose in best accommodating a local map (generated from sensor scan) in a partially developed global map. A Genetic algorithm is designed to search for the optimal robot pose which maximize the overall consistency of a map. In order to obtain the best result from genetic algorithm based search, domain specific knowledge is applied intelligently to generate an initial population. A fuzzy set theoretic approach is incorporated in this purpose to generate a sample based prediction of possible robot poses. The fuzzy logic system uses the prior knowledge about robot kinematics and its behavior at different environments to define a fuzzy regions to search for robot pose. The proposed algorithm is incremental in nature as opposed to batch algorithm in which the entire data set is processed all together, and therefore, has the great benefit to use for online robotic mapping.

Validity of the algorithm is tested by several experiments carried in simulated and real world indoor environments. Maps generated by the algorithm are topologically consistent and accurate for use in robot navigation.

Acknowledgements

I would like to thank my supervisors, Dr. Raymond Gosine and Dr. George Mann, for giving me the latitude to explore this research topic. I would specially like to thank Dr. George Mann for his help, guidance, feedback and inspiration necessary to take this research to its conclusion. Many thanks to the other members of the Intelligent system group in C-CORE for all the insightful conversations, specially to Lori for many thoughtful comments about the draft of this thesis. I am grateful to the Natural Science and Engineering Research Council of Canada (NSERC), Memorial University and C-CORE for financial support to carry out this research. The laboratory infrastructure assistance from Canada Foundation for Innovation Grants (CFI) New opportunity program and Memorial University of Newfoundland is greatly acknowledged. Finally, I would like to thank my husband Rajib, for his support and encouragement. I could not have done this without him.

Contents

List of Tables

List of Figures

List of symbols

$\mathcal{C}$ = Feasible region

c_n = Number of sample per unit volume of uncertainty

d_t = Total distance traveled by the robot at time t

F_t = Surface type within the time interval (t-1,t]

$\mathcal{F}_\mu$ = Fitness of $\mathbf{x}_t^\mu$

$\mathcal{F}(f(\mathbf{x}_{0:t}, \mathbf{s}_{0:t}))$ = Fitness of $\mathbf{x}_t$ in maximizing the consistency of $\mathbf{m}_t$

K = Manufacturing/assembling errors and sensor resolution parameters of a robot

$\mathbb{K}$ = Kalman gain

l_p = Population size

$\mathbf{m}_t$ = Global map generated at time t

$\mathbf{m}_t^{[i]}$ = i^{th} map generated at i^{th} iteration of EM algorithm

$\mathbf{m}_t^*$ = Maximum likelihood map at time t

$\hat{m}_t$ = Local map generated at time t

m = Number of islands in a population

N_{sample} = Number of samples in $\hat{\mathbf{X}}_t$

$P_t(n)$ = Population at the n^{th} generation of the GA at time t

$p(x)$ = Probability of x

$p(x|y,z)$ = Probability of x given y and z

$p^{\mathbf{x}_t^\mu(n)}$ = Probability of $\mathbf{x}_t^\mu(n)$ to be selected

$R(\theta_t)$ = Rotation matrix

$\mathbb{R}^k$ = k- dimensional real space

$\mathbb{R}$ = One- dimensional real space

r_{s_t} = Radius of uncertainty in spatial position at time t

r_{a_t} = Radius of uncertainty in orientation at time t

$\mathbf{s}_{0:t}$ = Laser measurements upto time t, $\{\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_t\}$

$\mathbf{s}_t$ = Exteroceptive sensor reading at time t (Laser sensor measurement)

$\mathbf{S}_t$ = Full State vector at time t composed of robot pose and map

$\mathbb{T}$ = Homogenous transformation matrix

$\mathbf{u}_t$ = Proprioceptive sensor reading at time t (Odometry)

$\mathbf{u}_{0:t}$ = Odometry readings upto time t, $\{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_t\}$

u_r = Mutation rate

v_t = Robot's linear velocity at within the time interval [t-1,t)

$\mathbf{x}_t$ = Robot pose at time t

$\tilde{\mathbf{x}}_t$ = The best estimate of the robot pose at time t

$\mathbf{x}_t^*$ = Maximum likelihood pose at time t

$\mathbf{x}_t^{\mu}(n)$ = The μ^{th} member of the n^{th} generation of the GA at time t

$\mathbf{x}_t^{j_i}$ = The i^{th} bit of $\mathbf{x}_t^j$

X = A set of M x-positions, $\{x_1, x_2, \ldots, x_M\}$

Y = A set of M y-positions, $\{y_1, y_2, \ldots, y_M\}$

$\underline{z}$ = Lower bound of z

$\overline{z}$ = Upper bound of z

ϵ, ϵ' = Parameters of mutation.

$\tilde{\phi}$ = A non-linear fuzzy mapping from the robot's odometry to the robot's pose

μ = Mean of a Gaussian

Σ = Normalized covariance matrix

η = Normalizer constant in Bayes rule

ω_t = Robot's angular velocity at within the time interval [t-1,t)

λ = A binary variable chosen randomly during crossover

μ_s = Space mutation array

μ_a = Angle mutation array

Θ = A set of N orientations, $\{\theta_1, \theta_2, \ldots, \theta_N\}$

s.t = such that

subscripts

t = Discrete time index

τ = Variable of integration with respect to time

superscripts

μ = Different members of a population

T = Transpose of a vector or matrix

List of acronyms

KF	=	Kalman Filter
EKF	=	Extended Kalman Filter
EM	=	Expectation Maximization
ML	=	Maximum Likelihood
CML	=	Concurrent Mapping and Localization
SLAM	=	Simultaneous Localization and Mapping
GA	=	Genetic Algorithm
SIR	=	Sequential Importance Resampling
RGO	=	Restricted Genetic Optimization

Chapter 1

Introduction

About this chapter. This chapter first introduces the problem addressed by this thesis. The scientific context of the problem is then described. This follows a summery of contributions to robotic mapping made by this research and finally the organization of the thesis.

1.1 Motivation

Researchers today have been focusing on the use of mobile robots for automation, particularly to replace human operated vehicles in harsh environment. This is because mobile robots have the potential to carry out tasks which are considered undesirable or difficult for humans due to hazardous working conditions (nuclear reactors, abandoned mines, etc.) or a shortage of skilled labor (health care). Other reasons for using robots include freeing human labor from menial and repetitive work (transportation and domestic service), increasing safety and reliability by augmenting human labor with robot assistance (inspection and surveillance), increasing productivity (farming and mining), and applications in education and leisure (tour guide and toys). Application of mobile robots have already shown significant success for exploring volcanoes [1], going places that are too dangerous for human access (e.g. abandoned mine) [2], searching for meteorites in Antarctica [3, 4], traversing desert [5], exploring and mapping sea bed [6], and even in exploring other

planets [7].

In order to carry out such high level tasks, an autonomous robot must possesses the fundamental knowledge of the working environment, in general, a map. The map can either be installed in the robot prior to operation or can be built online, depending upon the task it is performing. The mapping capability of an unknown environment allows a robot to be deployed with minimal infrastructure. When a map is generated online, the complexity of the mapping process increases due to the induced problem of robot localization with respect to the growing map.

The motivation of this research is to develop a robust algorithm for robotic mapping which enables the mobile robot to work in an environment without having any pre-installed knowledge of it. The world model required for high-level task planning will be acquired independently by the robot itself. In order to cope with working in an unknown world, the algorithm is devised in such a way that it requires no modification of the environment in order to facilitate a robust tool for robot localization.

1.2 Robotic mapping

Robotic mapping is the process of generating a spatial representation of a given environment from a series of sensor readings observed by a robot while traveling through that environment. This is generally regarded as one of the important problems in the pursuit of building truly autonomous robots [8]. The robotic mapping problem comes at varying degree of difficulty. In the most basic case, the mobile robot has access to a global positioning system (GPS)/differential GPS which provides it with almost accurate pose information. The problem of 'mapping with known robot pose' [9, 10, 11] is a trivial problem as compared to the general problem of robotic mapping [12] where reliable pose information is unavailable. When GPS is unavailable, as is the case of indoor, underground or underwater, the mobile robot inevitably accrue pose errors during mapping. This pose error induces a proportionate error in the map. The robotic mapping generally addresses the problem of map acquisition without reliable pose information.

There exist a cyclic nature in robotic mapping problem: while operating in an unknown environment, a fully autonomous robot needs to know its location in order to build a map of the uncharted territory, but to know its location, the robot needs a map. This cyclic nature of robotic mapping problem is illustrated in Figure 1.1. The robot has to

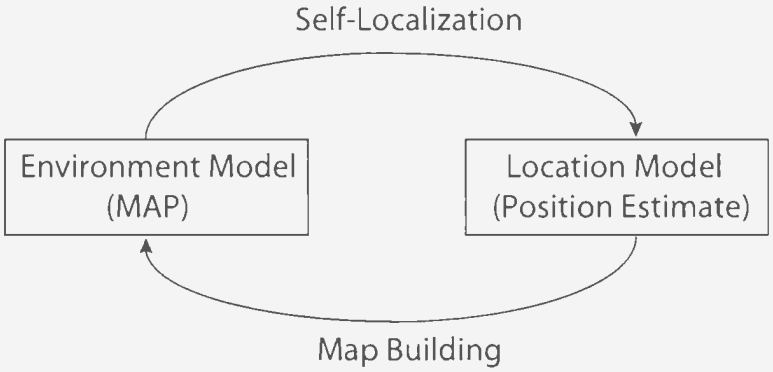


Figure 1.1: The problem of robotic mapping

simultaneously maintain two representations: firstly, an environment model or map, and secondly, a location model or position estimate. It must be able to run two perceptual processes, namely map-building and self-localization, simultaneously. This phenomenon of robotic mapping is often referred to as Concurrent Mapping and Localization (CML) [13] or Simultaneous Localization and Mapping (SLAM) [14] in robotic literature. In rest of this thesis, the words 'robotic mapping' , 'SLAM' and 'CML' will be used interchangeably.

1.3 Map representation

A great number of mobile robot systems [15, 16, 17, 18] in literature rely on maps for navigation. The robot can represent the map of an environment in a number of ways depending on the type of the navigational task. For indoor environments, there are two fundamental paradigms of mapping: the *grid-based (metric)* paradigm and the *topological* paradigm [19].

- The grid-based paradigm represents environment by evenly-spaced grids. Each grid cell may, for example, indicate the presence of an obstacle in the corresponding

region of the environment. Grid-based maps are relatively easy to construct but require considerable storage and can be computationally demanding [12].

- The *topological* paradigm represents robot environment by directed graph. Nodes in such graphs corresponds to distinct places, or landmarks (such as doorways), while edges in the graphs represent connected path between two landmarks if there exist any direct path between them in the real world. Topological maps are difficult to construct as they have to provide geometric relationships between the observed landmarks. However, this approach of map building is less sensitive to the robot's odometry error as opposed to grid-based approach [12].

Based on the coordinate system the map is presented, maps are divided into two category: *world-centric* map and *robot-centric* map. World-centric maps are represented in a global coordinate space. Robot-centric map, on the other hand, are described in measurement space. They describe sensor measurements a robot would receive at different locations[12].

This research focuses on developing an algorithm for generating world-centric, metric map of environment. Metric map is chosen over topological map as it does not require feature (landmarks) extraction from sensor data. Furthermore, it does not require engineering of featureless environment by placing artificial landmarks. Similarly, world-centric map is chosen over robot-centric map as the world-centric map facilitates robot navigation more than robot-centric map.

1.4 The challenges in robotic mapping

There exists a number of factors which impose practical limitations on a robot's ability to acquire a consistent map of an environment while localizing itself with respect to that evolving map. A comprehensive study on these factors are available in [12]. The key challenges of robotic mapping are briefly discussed below.

1.4.1 Sensor and sensor noise

To acquire a map, a robot must possesses sensors that enable it to perceive the outside world. The mobile robots having mapping capability are usually equipped with two type of sensors [20]:

- Proprioceptive - these sensors measure robot movement relative to the robot's own frame of reference. e.g. optical wheel encoders, Doppler sensors, gyroscopes.
- Exteroceptive - these sensors measure the layout of the environment relative to the robot's frame of reference e.g. range sensors (e.g. laser range finder, sonar), vision using camera.

In general, sensors are subject to errors, often referred to as measurement error or sensor noise [12, 21]. A key challenge in robotic mapping arises from the statistically dependent nature of different measurement noises [12]. For example, the control commands (obtained from optical encoder reading) issued during environment exploration carry important information for building maps, since they convey information about the locations at which different sensor measurements were taken. But the robot's motion is also subjected to errors. The errors in optical encoder reading (generally known as odometry) usually arise from various kinematics characteristics of the robot and also from varying amount of slippage and skidding at different terrain conditions. The odometry errors, once introduced, accumulate unboundedly over time and it effects the interpretation of the future sensor measurements. Therefore, odometry alone is insufficient to determine a robot's pose (location and orientation) relative to its environment.

1.4.2 Data association or correspondence problem

The data association or correspondence problem [22, 23] refers to the problem of determining whether the sensor measurements taken at different points in time correspond to the same physical object in the world. This is one of the most challenging problem of robotic mapping [12]. Many robotic mapping algorithms perform with an underlying assumption

of perfect data association between different features in environment [21, 14]. However, solving data association problem is an active research in robotic mapping [24, 25, 26]. A very well-known data association problem is the loop closing problem while mapping cyclic environment. This problem is complicated due to the fact that at the time of cycle closing, the robot's accumulated pose errors might be unboundedly large and the robot might fail to establish perfect association between features.

1.4.3 Real-time requirements

To execute high level tasks in real-time, the robot should be capable of building a map online. This time requirement often demands that the underlying algorithm must be incremental and sufficiently simple to be performed online. In addition, this developed map must be easily accessible. For example, accurate fine-grained CAD models are often inappropriate to use by a self-navigating robot which takes action in real time.

1.4.4 Complexity and dynamics of environment

Environments that are complex (each entity in the environment might possesses several dimensions) and dynamic (real world environment is constantly changing over time, small or large), pose a great challenge for the robot to maintain an exact environment model. Practically, there are almost no mapping algorithms that can learn meaningful maps of dynamic environments [12]. Rather, most of the existing mapping algorithms follow a static world assumption. In other words, they are applied in relatively short time windows during which the respective environments are static.

The proposed research addresses some of the above mentioned challenges in robotic mapping. It combines intelligent computing techniques, namely fuzzy logic and genetic algorithm, to propose a solution to CML of mobile robot.

1.5 Intelligent computing for robotic mapping

Robotic mapping is a complex problem partly because of sensor noises. Whatever decision a robot infers about its environment or its own pose, the decision is always plagued with errors. Assumption of various stochastic models (e.g. Gaussian) to accommodate these errors performs well for certain type of exteroceptive sensors (e.g. laser) [27, 21]. Errors in odometry, on the other hand, are not suitable to be modeled stochastically [28]. Several real world parameters(e.g. amount of slippage and skidding) have non-Gaussian relationship with the growth of odometry errors [29]. Therefore, stochastic modeling of odometry error may produce unreliable result. Fuzzy logic provides a natural framework to define and solve qualitative (as well as quantitative) relations between various quantities. Therefore, a fuzzy logic based approach is proposed in this research to model the odometry error of mobile robot.

Application of sample based algorithms (e.g, particle filter) have produced significant results in robotic mapping [24, 25, 26. 30, 31]. The effectiveness of sample based algorithm lies in the capability to accommodate any arbitrary uncertainty in sensor measurements. Moreover, sample based algorithms offer effective solution to the data association problem. Genetic algorithms (GAs) are a class of sample based algorithms developed on the principle of natural evolution. GAs have potential application in robotic mapping. The variation including operators of GAs (mutation and crossover) have the capacity to generate samples based on their fitness. Theory of particle filters, on the other hand, only allow resampling over the existing sample set. Moreover, the property of natural selection in supporting better performing individuals to survive offers an iterative solution to data association problem of robotic mapping. The proposed research combines fuzzy logic and a GA to develop a robust solution to the robotic mapping problem.

1.6 Contributions to robotic mapping

The contribution of this research to the CML of mobile robot lies in the following aspects:

- It provides a fuzzy framework for modeling the errors in mobile robot's odometry.
- It introduces a new sample based method for solving CML. As opposed to the well known sample based method for CML, *particle filter*[21, 25, 26, 30, 31], the proposed method does not keep on updating the history of a set of samples upon receiving new sensor measurement. Rather, it generates new samples through the use of genetic operators (crossover and mutation). This removes the constraints on the initial sample distribution to closely resemble the original distribution.
- It combines two soft computing methods for solving CML of mobile robot. This is completely a new concept in the literature of robotic mapping.
- Finally, it provides a theoretical frame work of CML as an optimization problem.

1.7 Thesis outline

This thesis will present a novel algorithm for CML of mobile robot. Experiments on simulated and real world data set will be provided to validate the performance of the proposed algorithm. The rest of the thesis is organized as follows:

Chapter 2 discusses some of the state-of-art algorithms in robotic mapping.

Chapter 3 describes the proposed algorithm for CML.

Chapter 4 shows experimental results to validate the proposed research in various simulated and real-world indoor environments.

Chapter 5 provides concluding discussion about the research and indicates some future works.

Chapter 2

Literature Review

About this chapter. This chapter reviews the literature on Robotic mapping and intends to provide the current trends in CML. First, existing mapping algorithms are classified into different categories based on underlying principle. It follows a general discussion about algorithms in each category along with their advantages and shortcomings.

2.1 History of robotic mapping

Robotic mapping has been an active research topic since 1980. In the 1980s, a popular work of Elfes [9] on metric mapping resulted the *Occupancy grid algorithm* [10, 11] which represents the map by fine-grained grids to model the occupied and free spaces of the environment. This algorithm experienced great popularity and has been used in a number of robotic systems [32, 33, 16, 34, 35, 36]. The *Occupancy grid algorithm* has the capability to handle uncertainty in exteroceptive sensor (e.g. sonar) using probabilistic techniques. However, this algorithm requires exact robot pose information for mapping. In other words, occupancy grid mapping is 'mapping with known pose'. The general problem of robotic mapping in simultaneously recovering a map as well as the robot pose was first solved by Smith *et.al* [37, 27] in 1990. This seminal work resulted in a new research area in mobile robotics for simultaneously solving the mapping problem and the problem of localizing the robot relative to the growing map. Since then, robotic mapping has commonly been

referred to as SLAM or CML.

Robotic mapping is inherently challenged by measurement uncertainty. As discussed in Section 1.4.1, perceptual noise is complex and exhibits random nature. This results in adopting probabilistic techniques for solving robotic mapping. Probabilistic techniques approach the problem by explicitly modeling different sources of noise and their effects on the measurements. Virtually all state-of-the-art mapping algorithms in robotics literature are probabilistic. Probabilistic algorithms employ probabilistic models of robot's pose and environment, and rely on probabilistic inference for turning sensor measurements into a map. Some algorithms [13, 38, 30, 14, 39] make the probabilistic thinking very explicit by providing mathematical derivations of the algorithms from probabilistic principles. Others [40, 41, 42] use techniques that on the surface do not look specifically probabilistic, but in fact can be interpreted as probabilistic inference under constraints [12]. Thrun [12] provides an extensive survey on the probabilistic algorithms in robotic mapping.

The existing state-of-the-art robotic mapping algorithms can be classified into four categories based on the underlying theory:

1. Kalman Filter based approach.
2. Expectation Maximization (EM) based approach.
3. Particle filter based approach.
4. Genetic algorithm based approach.

Algorithms in each category have their own advantages and shortcomings. A brief discussion about each category is given below.

2.2 Kalman Filter based approach

Kalman filter (KF) [43, 44] is a form of Bayesian filters [45] in which the posterior over system states is explicitly represented as unimodal Gaussian distribution. The general

form of Bayes filter in the robotic mapping problem can be stated as [12],

$$p(\mathbf{x}_t, \mathbf{m_t}|\mathbf{s}_{0:t}, \mathbf{u}_{0:t}) = \eta p(\mathbf{s}_t|\mathbf{x}_t, \mathbf{m_t}) \int p(\mathbf{x}_t|\mathbf{u}_t, \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}, \mathbf{m_t}|\mathbf{s}_{0:t-1}, \mathbf{u}_{0:t-1}) d\mathbf{x}_{t-1} \quad (2.1)$$

Here,

$$\begin{aligned} \text{subscript } t &= \text{discrete time index} \\ \mathbf{x}_t &= \text{robot pose} \\ \mathbf{m}_t &= \text{global map generated at time } t \\ \mathbf{s}_t &= \text{exteroceptive sensor measurement (e.g. range measurement)} \\ \mathbf{u}_t &= \text{proprioceptive sensor reading (odometry)} \\ \eta &= \text{normalizer constant} \\ \mathbf{u}_{0:t} &= \{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_t\} \end{aligned}$$

The distribution $p(\mathbf{x}_t, \mathbf{m_t}|\mathbf{s}_{0:t}, \mathbf{u}_{0:t})$ is read as the joint posterior over robot pose and map conditioned both on the exteroceptive and proprioceptive sensor measurements. The generative distribution $p(\mathbf{s}_t|\mathbf{x}_t, \mathbf{m_t})$ probabilistically describes how sensor measurements $\mathbf{s}_t$ are generated for different poses and maps. Therefore, $p(\mathbf{s}_t|\mathbf{x}_t, \mathbf{m_t})$ is often referred to as the *perceptual model* [33, 16, 13, 31, 30, 46, 6, 47] in robotics. Similarly, the generative distribution $p(\mathbf{x}_t|\mathbf{u}_t, \mathbf{x}_{t-1})$ in (2.1) specifies the effect of the control $\mathbf{u}_t$ (the amount of commanded odometry move) on the robot pose $\mathbf{x}_t$. It describes the probability that the control $\mathbf{u}_t$, if executed at the world state $\mathbf{x}_{t-1}$, leads to the state $\mathbf{x}_t$. For moving robots, the probability $p(\mathbf{x}_t|\mathbf{u}_t, \mathbf{x}_{t-1})$ is usually referred to as the *motion model* [33, 16, 13, 31, 30, 46, 6, 47].

The classical approach of solving SLAM/CML using KF was first introduced by Smith *et.al* [37, 27]. This original work proposes a KF-based statistical framework to solve the robotic mapping problem. Motivated by this work, several researchers [21, 14, 48, 49, 6, 50, 39] followed the KF based approach to solve the SLAM problem.

In KF-based algorithms, the posterior over full state vector is represented with Gaussian. In the context of robotic mapping, the full state vector $\mathbf{S}_t$ comprises the robot's pose $\mathbf{x}_t$ and the map $\mathbf{m}_t$:

$$\mathbf{S}_t = (\mathbf{x}_t, \mathbf{m}_t). \quad (2.2)$$

The probability $p(\mathbf{x}_t, \mathbf{m}_t | \mathbf{s}_{0:t}, \mathbf{u}_{0:t})$ in (2.1) is represented by Gaussian. For robots operating on a planar surface, the robot's pose $\mathbf{x}_t$ is usually modeled by three variables: the cartesian coordinates in the plane, (x, y), and the heading direction, θ. Maps in KF-based algorithms are commonly represented by the Cartesian coordinates of a set of features. Appropriate features may be landmarks, distinctive objects or shapes in the environment. Denoting the number of feature in the map by N, the state vector defined in (2.2) is given by the following $(2N + 3)$-dimensional vector:

$$\mathbf{S}_t = (x, y, \theta, m_{1,x,t}, m_{1,y,t}, m_{2,x,t}, m_{2,y,t}, \ldots, m_{N,x,t}, m_{N,y,t}) \tag{2.3}$$

Here $m_{N,x,t}, m_{N,y,t}$ are the Cartesian coordinates of the N-th feature in the map $\mathbf{m}_t$. The mean and the covariance matrix, μ_t and Σ_t, respectively, of the Gaussian representing the probability $p(\mathbf{x}_t, \mathbf{m_t} | \mathbf{s}_{0:t}, \mathbf{u}_{0:t})$ are of dimension $2N + 3$ and $(2N + 3)^2$, respectively.

2.2.1 Key advantages of Kalman filter in robotic mapping

In robotic mapping, the primary advantage of KF is that it estimates the full posterior over maps and poses online. To date, the algorithms that are capable of estimating such full posterior are based on KF or extensions thereof. In addition to the most likely map and robot poses, KF maintains the full uncertainty in the map, which can be highly beneficial when using the map for navigation. Additionally, the KF based approach can be shown to converge, with probability equal to one, to the true map and the robot pose, up to a residual uncertainty distribution that largely stems from an initial random drift [21, 50].

2.2.2 Limitations of Kalman filter in robotic mapping

The KF suffers from some serious limitations in the context of robotic mapping. KF-based mapping relies on three basic assumptions:

1. The motion model (next state function, in theory of KF) must be linear with added Gaussian noise.

2. The perceptual model must be linear with added Gaussian noise.

3. The initial uncertainty in robot's pose must be Gaussian.

The most important limitation of the KF approach lies in the Gaussian noise assumption. In particular, the assumption that the measurement noise must be independent and Gaussian poses a key limitation with important implications for practical implementations. This makes KF based approaches unable to cope with the correspondence problem [23, 22] (the problem of associating individual sensor measurements with features in the map). Therefore, the practical implementations of the KF based approaches usually require a sparse set of features which are sufficiently distinctive, either by their measurement characteristics or by their locations, so that they can be distinctly identified [26, 24]. Error in the identification of features in an environment usually implies a failure of the mapping algorithm. For this reason, KF based approaches are usually forced to ignore large portions of the sensor data, and work with a small number of landmark-type features only [12]. The resulting maps contain the locations of these landmarks but usually lack detailed geometric descriptions of the environment.

The second limitation of KF stems from the assumption of linear motion and perceptual models [12]. In a linear motion model, the robot pose $\mathbf{x}_t$ and the map $\mathbf{m}_t$ depend linearly on the previous pose $\mathbf{x}_{t-1}$ and map $\mathbf{m}_{t-1}$, and also linearly on the control $\mathbf{u}_t$. This is trivially the case for the map since according to the static world assumption (also known as, *Markov assumption*), the map does not change. However, the pose $\mathbf{x}_t$ is usually governed by a nonlinear trigonometric function that depends nonlinearly on the previous pose $\mathbf{x}_{t-1}$ and the control $\mathbf{u}_t$ [28]. Besides, sensor measurements in robotics are usually nonlinear, with non-Gaussian noise. To accommodate such nonlinearity, an extension on the basic KF has been proposed. This extension of KF, known as *Extended Kalman Filter*, has been used by several researchers for robotic mapping [11, 39].

The Extended Kalman Filter (EKF) approximates the robot's nonlinear motion model using a linear function obtained via the Taylor series expansion. Single motion commands are often approximated by a series of much smaller motion segments, to account for nonlinearities. For most robotic vehicles, such an approximation works well.

2.3 Expectation Maximization algorithm based approach

Expectation Maximization (EM) is a statistical algorithm which was developed in the context of maximum likelihood (ML) estimation with latent variables, in an influential paper by Dempster *et al.* [51]. Application of this algorithm in robotic mapping has produced significant results [52, 13, 53].

In robotic mapping, the EM algorithm iterates in two steps

1. An *expectation step* or E-step. Here, the posterior over robot poses is calculated for a given map.

2. A *maximization step* or M-step. Here, the most likely map is calculated given the pose expectations.

The function that is being maximized is the expectation over the joint log likelihood of the sensor measurements $\mathbf{s}_{0:t}$ and the robot's trajectory $\mathbf{x}_{0:t}$

$$\mathbf{m}_t^{[i+1]} = \arg\max_{\mathbf{m}_t} E_{\mathbf{x}_{0:t}}[\log p(\mathbf{s}_{0:t}, \mathbf{x}_{0:t}|\mathbf{m}_t)|\mathbf{m}_t^{[i]}, \mathbf{s}_{0:t}] \tag{2.4}$$

Here, $\mathbf{m}_t^{[i]}$ is the map generated at i^{th} iteration of the EM algorithm. Part of the likelihood that is being maximized is the robot's path $\mathbf{x}_{0:t}$. However, in robotic mapping the path is unknown. Therefore, (2.4) computes the expectation of this likelihood over all possible paths the robot may have taken. Under few assumptions, (2.4) can be re-expressed as the following integral [12]

$$\mathbf{m}_t^{[i+1]} = \arg\max_{\mathbf{m}_t} \sum_{\tau} \int p(\mathbf{x}_\tau|\mathbf{m}_t^{[i]}, \mathbf{s}_{0:t}) \log p(\mathbf{s}_\tau|\mathbf{x}_\tau, \mathbf{m}_t) d\mathbf{x}_\tau \tag{2.5}$$

Here, τ is the variable of integration with respect to time. The E-step of the EM-based robotic mapping algorithm calculates $p(\mathbf{x}_\tau|\mathbf{m}_t^{[i]}, \mathbf{s}_{0:t})$ which is the posterior for the pose $\mathbf{x}_\tau$ conditioned on the data $\mathbf{s}_{0:t}$ and the i-th map $\mathbf{m}_t^{[i]}$. To calculate this pose posterior at time τ, $\mathbf{x}_\tau$, sensor measurements in the entire time interval $\{1, \ldots, t\}$ is used, even for $\tau < t$. Thus, it is required to incorporate both past and future data relative to the time step τ for posterior estimation over robot's poses.

The goal of the M-step is to find a new map $\mathbf{m}_t$ that maximizes the log likelihood of the sensor measurements $\log p(\mathbf{s}_\tau|\mathbf{x}_\tau)$ for all τ and all poses $\mathbf{x}_{0:t}$ and under the expectation $p(\mathbf{x}_\tau|\mathbf{m}_t^{[i]}, \mathbf{s}_{0:t})$ calculated in the E-step. The algorithms proposed in [13] and [52] solve this high-dimensional maximization problem by considering each map location (x, y) independently. This approach of solution assumes that the map is represented by a finite number of locations, e.g., by a fine-grained grid. The component-wise maximization is then relatively straightforward.

2.3.1 Advantages and shortcomings of EM-based robotic mapping algorithms

The key advantage of the EM algorithm lies in the fact that it solves the correspondence problem. It does so by repeatedly re-localizing the robot relative to the present map in the E-step. The pose posteriors calculated in the E-step correspond to different hypotheses as to where the robot might have been, and hence imply different correspondences. By building maps in the M-step, these correspondences are translated into features in the map, which then either get reinforced in the next E-step or gradually disappear. The capability of handling correspondence problem makes EM-based algorithms superior over KF/EKF based techniques. Besides, EM algorithms are capable of generating consistent maps of large-scale cyclic environment even if all features look alike and cannot be distinguished perceptually (as discussed in Section 2.2.2, KF/EKF can not take care of such situations). However, EM based algorithms do not retain a full notion of uncertainty. Instead, these algorithms perform hill climbing in the space of all maps, in an attempt to find the most likely map. To achieve this task, EM-based algorithms require processing the entire data multiple times. In other words, these algorithms are batch algorithm and cannot generate map incrementally. In this aspect, EM is inferior to KF/EKF because most KF/EKF-based algorithms are capable of generating the map online.

A new family of algorithm has developed in literature which attempts to combine the advantages of both KF/EKF and EM algorithms. This new technique, known as

Incremental maximum likelihood approach, combines the incremental mapping quality of the KF/EKF as well as the power of EM in providing solution to correspondence problem.

2.3.2 Incremental maximum likelihood method for CML

The objective of the incremental Maximum likelihood method [11, 38, 54, 55] in robotic mapping is to incrementally build a single map as the new sensor data arrives, but without keeping track of any uncertainty in pose and map. Such a methodology can be viewed as a M-step in EM, without an E-step [12]. The main advantage of this algorithm is its simplicity.

Mathematically, the basic idea is to maintain a series of maximum likelihood maps $(\mathbf{m}_1^*, \mathbf{m}_2^*, \ldots,)$ along with a series of maximum likelihood poses $(\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots,)$. The t^{th} map and pose are constructed from the $(t-1)^{th}$ map and pose via maximization of the marginal likelihood given by

$$< \mathbf{m}_t^*, \mathbf{x}_t^* >= \arg\max_{\mathbf{m}_t, \mathbf{x}_t} p(\mathbf{s}_t|\mathbf{x}_t, \mathbf{m}_t) p(\mathbf{x}_t, \mathbf{m}_t|\mathbf{u}_t, \mathbf{x}_{t-1}^*, \mathbf{m}_{t-1}^*). \tag{2.6}$$

The map $\mathbf{m}_t$ can be uniquely determined if the pose $\mathbf{x}_t$ is known. The incremental maximum likelihood method simply requires a search in the space of all poses $\mathbf{s}_t$ when a new data item arrives which in turn is used to determine the pose $\mathbf{x}_t^*$ that maximizes the marginal posterior likelihood. Similar to KF, this approach can generate a map in real-time, though without maintaining a notion of uncertainty. Similar to EM, it maximizes likelihood.

In incremental maximum likelihood method, once a pose $\mathbf{x}_t^*$ and a map $\mathbf{m}_t^*$ have been determined, they cannot be revised based on the future data. The KF based algorithms are also characterized by the same feature. This weakness reveals itself in the inability to map cyclic environments, where the errors in the poses may accumulate unboundedly.

2.4 Particle filter based robotic mapping

Particle filters are the most recent alternative of KF in the context of robotic mapping [30, 56]. The key idea of using particle filter is to represent the posterior distribution by a random collection of weighted particles which approximate the desired distribution [57, 58]. As the posterior is not approximated in parametric form (as opposed to KF/EKF), particle filter can accommodate almost arbitrary sensor characteristics, motion dynamics and noise distribution. In the literature of robotic mapping, there exist a number of algorithms which use particle filter. Hybrid approach and FastSLAM, two popular algorithms in this category, are discussed in this section.

2.4.1 Hybrid approach

The hybrid approach [12] of CML [31, 47] estimates the posterior over the robot's poses using particle filter. These algorithms [31, 47] use the incremental maximum likelihood approach to build map while maintaining a posterior distribution over the robot poses. This distribution is calculated using the standard Bayes filter applied only to robot poses

$$p(\mathbf{x}_t|\mathbf{s}_{0:t}, \mathbf{u}_{0:t}) = \eta p(\mathbf{s}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{u}_t, \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{s}_{0:t-1}, \mathbf{u}_{0:t-1}) d\mathbf{x}_{t-1} \tag{2.7}$$

By retaining a notion of the robot's pose uncertainty, conflicts encountered during mapping large cyclic environment can be identified, and the appropriate corrective action can be taken. A particle filter is employed to calculate the pose posterior $p(\mathbf{x}_t|\mathbf{s}_{0:t}, \mathbf{u}_{0:t})$.

Unlike the incremental maximum likelihood method, the hybrid approach has the ability to correct the map backward in time whenever an inconsistency is detected.

Limitations of hybrid approach

The hybrid approach suffers many deficiencies. First and foremost, the decision to change the map backwards in time is discrete which, if wrong, can lead to a catastrophic failure [12]. Moreover, the approach cannot cope with complex ambiguities, such as the uncertainty that arises when the robot traverses multiple nested cycles. Finally, the hybrid

approach is, strictly speaking, not a real-time algorithm, since the time it takes to correct a loop depends on the size of the loop. However, practical implementations [59, 31, 17] appear to work well in real-time when used in office-building type environments.

2.4.2 FastSLAM

FastSLAM [24, 25, 26, 60] is a SLAM algorithm that integrates both particle filter and EKF. It exploits a structural property of the SLAM problem that feature estimates are conditionally independent given the robot path. More specifically, correlations in the uncertainty among different map features arise only through the robot's pose uncertainty. If the robot was told its correct path, the errors in its feature estimates would be independent of each other. This fact allows to define a factored representation of the posterior over poses and maps. FastSLAM implements such a factored representation, using particle filters for estimating the robot path (not the robot pose). Conditioned on these particles the individual map errors are independent, hence the mapping problem can be factored into separate problems, one for each feature in the map. FastSLAM estimates these feature locations by EKF. The basic algorithm can be implemented in time logarithmic in the number of landmarks, using efficient tree representations of the map [24]. Hence, FastSLAM offers computational advantages over plain EKF implementations and many of its descendants.

The key advantage of FastSLAM is the fact that data association decisions can be made on a per-particle basis. As a result, the filter maintains posteriors over multiple data associations, not just the most likely one. This feature makes FastSLAM significantly more robust to data association problems than algorithms based on maximum likelihood data association [61]. A final advantage of FastSLAM over EKF-style approaches arises from the fact that particle filters can cope with non-linear robot motion models, whereas EKF-style techniques approximate such models via linear functions [61].

Limitations of FastSLAM

The main limitation of FastSLAM is the fact that it maintains dependencies in the estimates of feature locations only implicitly, through the diversity of its particle set [61]. This disadvantage is also the source of FastSLAM's efficiency - a key advantage of FastSLAM over previous techniques. In certain environments this fact can negatively effect the convergence speed when compared to the mathematically more cumbersome EKF [14, 21].

2.4.3 DP-SLAM

A recent development in particle filter based robotic mapping is Distributed Particle-Simultaneous Localization And Mapping (DP-SLAM) [62][63]. DP-SLAM does not require feature extraction or identification from sensor data. It provides an elegant solution to efficiently store the individual maps assigned to different particles. The core difference between DP-SLAM and FastSLAM lies in the way of representing world state. Rather than using KF on landmark positions (like FastSLAM), the DP-SLAM uses probabilistic occupancy maps. Unlike the conventional occupancy map [9], each grid in the occupany map of DP-SLAM is actually a tree containing observations for different particles. The map representation technique which is termed as Distributed Particle mapping in [62][63] enables the algorithm to maintain and update hundreds and thousands of candidate robot poses and maps in real time as the robot moves through the environment.

Limitations of DP-SLAM

The price for the efficiency in memory utilization (as proposed in DP-SLAM) is that the data retrieving from a grid cell is far more complicated than a simple array access. Besides, the DP-SLAM algorithm requires large number of particles for closing loops.

2.5 Genetic algorithm based approach

Application of Genetic algorithms (GAs) in robotic mapping is a relatively a newer concept. Its application in CML of mobile robot was first introduced by Ducket [64] in 2003, though

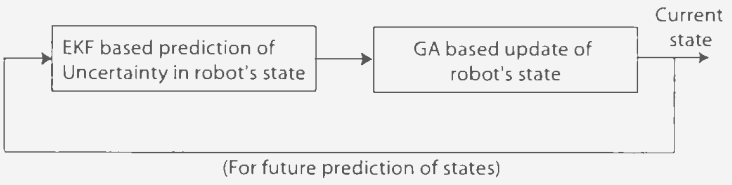


Figure 2.1: Restricted Genetic Optimization algorithm for mobile robot localization with respect to a given map

GAs have been successfully implemented in navigation [65, 66, 67] and path planning [68] of mobile robot since 1990s, .

In 2002, Luis *et. al.* [28, 69, 70] proposed algorithms for mobile robot localization with respect to a given map using ultrasonic sensors. These algorithms employ GA along with EKF to search for the optimal robot pose. The algorithms proposed in [28, 69] are known as Restricted Genetic Optimization (RGO) as GA is applied only in restricted areas of the solution space. The EKF generates a seed which is used to estimate a neighborhood where the true value of the robot pose is located. With this information, and inside this neighborhood the most accurate solution is searched. The search is performed employing a GA. Each chromosome of the RGO represents the difference with the best point of the last generation. New generations are oriented in the direction of the steepest slope of the cost function (steepest descend), and have smaller distance to the correct estimate at each generation. The selection process is cross-generational, and the fitness function is an approximation of the probability of a current solution conditioned on all the sensor measurements and a priori knowledge about initial state. The process of RGO can be viewed as shown in Figure 2.1. However, RGO is proposed for robot localization with respect to an *a priori* map.

The very first application of GA in robotic mapping [64] defines CML as a global optimization problem and employs a GA to search for the optimal solution. The trajectory generated by a robot during the complete process of data collection is divided into small segments (of 3m each). The robot's own measurements of its trajectory are used as a generative model. A GA is designed to search for a set of correction vectors associated

with the small segments of the robot's trajectory. The correction vectors for different trajectory-segments are chosen in such a way that the effect of modifying each small segment combinedly develop a compact and consistent map of the environment. Two heuristic functions, namely *Map Consistency* (MC_1) and *Map Compactness* (MC_2), are defined to measure the compactness and consistency of a map.

- *Map Consistency* (MC_1): A map is assumed to be consist of finer grid cells of resolution 10cm. For each grid cell i, two quantities are calculated: occ_i, the number of laser readings which indicate that the cell is occupied, and emp_i, the number of readings which indicated that the cell is empty. The Map compactness function tries to measure the degree of disagreement or 'conflict' between the sensor readings. The measure is calculated as

  $$MC_1 = \Sigma_i min(occ_i, emp_i), \tag{2.8}$$

  by taking the minimum of the occ_i and emp_i values for all cells i.

- *Map Compactness* (MC_2): This function tries to reward the GA for producing smaller, more compact maps. It does so by fitting a bounding box to the map that indicates the total area covered by the cells with $occ_i > 0$. The measure is calculated as

  $$MC_2 = (x_{max} - x_{min}) \times (y_{max} - y_{min}). \tag{2.9}$$

  where x_{max} and x_{min} refer to the maximum and minimum x-coordinates of the bounding box measured in the number of grid cells.

Linear combination of MC_1 and MC_1 defines a fitness function F.

$$F = MC_1 + wMC_1. \tag{2.10}$$

where the weight $w = 0.3$ determines the relative importance of the two heuristics in the fitness function.

2.5.1 Limitations of the existing GA-based CML

The GA-based CML algorithm proposed in [64] is subjected to certain limitations:

- The algorithm requires the entire data set, both odometry and laser measurements, for processing. Being a batch algorithm, it is unable to generate a map incrementally.
- There exist a number of cells in the grid map whose status are 'unknown' (neither empty, nor occupied). The proposed fitness function does not consider these cells while measuring the consistency of the map. This weakness manifest itself by producing topologically inconsistent map while maintaining a low value of fitness, specially in case of complex environments (e.g. turning of a corridor).
- The algorithm assumes that the odometry data are always within a fixed range ($\pm 2\%$) of the true values of robot poses. This hypothesis totally disagree with the mobile robot's odometry. As discussed in Section 1.4.1, the robot's odometry is subjected to various errors and the magnitude of the errors grow with time. Besides, skidding and slippage while traversing irregular terrain might introduce huge error in odometry. Therefore, the assumption about odometry to closely resemble the true robot pose imposes serious limitation on the applicability of the algorithm.
- The assumption of small odometry error makes the algorithm unable to cope with the huge errors that arise during loop-closure while mapping large cyclic environment.
- The algorithm does not provide necessary theoretical analysis of CML as an optimization problem.

2.6 Summary

Almost all existing state-of-the-art algorithms in robotic mapping are probabilistic. The KF and EKF based algorithms are devised only to handle Gaussian noise in sensor measurements which makes these algorithms unable to cope with the data association problem of robotic mapping. The EM based algorithms relaxes the Gaussian noise assumption in

sensor measurements and overcome the data association problem by performing a hill-climbing search in the space of all possible maps. In doing so, the EM based algorithms lose the capability to build map incrementally as the entire data set is required to search for the most likely map. Particle filter based algorithms has the capacity to generate map incrementally. These algorithms are also capable to cope with data association problem. Sample based algorithms like particle filter, are recent development in robotic mapping and has gained much popularity. The particle filter based algorithms approximate the original error distribution using a set of samples as opposed to the EM based algorithms where, parametric model of error distribution is used. However, particle filter still imposes the restriction on the sample distribution (known as proposal distribution, in particle filter theory) to closely resemble the original distribution. The GAs are another class of sample based algorithm currently being used in robotic mapping. The GAs are more flexible (than particle filter) in operating with samples in the context that they can re-parent samples while particle filters are only capable to do resampling. The application of GA in robotic mapping has not been fully matured and has reported only in a very limited number of articles.

The present research proposes a novel sample-based algorithm for robotic mapping using GA and fuzzy logic.

Chapter 3

Proposed algorithm for robotic mapping

About this chapter. This chapter describes the proposed robotic mapping algorithm. First, CML is formulated as an optimization problem. This follows the discussion on the proposed fuzzy logic based odometry error modeling technique. Finally, the genetic algorithm based global search process for robot pose is described.

3.1 Introduction

This research presents a novel sample based robotic mapping algorithm. The proposed algorithm formalizes CML as an optimization problem and develops a mapping algorithm using two soft computing methodologies, namely fuzzy logic and genetic algorithm. The function to be optimized is a measure of quality of a robot pose while accommodating a sensor scan in a partially developed global map of the environment. The domain of the robot pose within which the objective function tries to optimize itself is defined using a fuzzy inference system. This fuzzy rule based system models the errors in odometry of mobile robot and generates a fuzzy sample based prediction of the robot pose. The underlying fuzzy mapping rules infer the uncertainty in a robot pose after execution of each motion command. The rule base is constructed from expert knowledge of the robot's

kinematics and its behavior at different terrain conditions. The search for true robot pose from the fuzzy sample based prediction of uncertainty is performed by a genetic algorithm. The fitness function is set as the optimization function while the fuzzy sample based prediction of robot pose acts as the initial population for the proposed GA. The genetic operators are designed in such a way that they are capable of intelligently extending the search outside the initial fuzzy prediction. The property of *natural selection* (in favoring better performing individuals to survive) is utilized to refine the data associations proposed by samples in different generations.

The proposed algorithm process data in an incremental fashion, i.e. at any point in time only the available sensor measurements are utilized to generate a partial global map of the environment.

3.2 CML as a global optimization problem

3.2.1 Optimization problem

The Optimization problem is a family of computational problem where a solution having the minimum (or maximum) value of the objective function is searched in a feasible region. A Feasible region is a region in the solution space where all the constraints are satisfied. A global optimization problem can be specified, as suggested in [71], in the form

$$\min \varphi(z) \text{ or } \max \varphi(z)$$
$$\text{s.t. } z \in \mathbf{z},\ \zeta(z) \in \Phi. \tag{3.1}$$

Here,

$$\mathbf{z} = [\underline{z}, \overline{z}] = \{z \in \mathbb{R}^k | \underline{z} \leq z \leq \overline{z}\} \tag{3.2}$$

is a bounded or unbounded box in k- dimensional real space $\mathbb{R}^k$ with $\underline{z}$ as lower bound and $\overline{z}$ as upper bound of z. $\varphi : \mathbf{z} \rightarrow \mathbb{R}$ is a continuous objective function, $\zeta : \mathbf{z} \rightarrow \mathbb{R}^l$ is a vector of l continuous constraint functions $\zeta_1(z), \ldots, \zeta_l(z)$, and Φ is a box in $\mathbb{R}^l$ defining the constraints on $\zeta(z)$. The feasible region is defined as

$$\mathcal{C} = \{z \in \mathbf{z} | \zeta(z) \in \Phi\}. \tag{3.3}$$

A global solution to the optimization problem is a feasible point $\hat{z} \in \mathcal{C}$ such that

$$\varphi(\hat{z}) = \min_{z \in \mathcal{C}} \varphi(z) \text{ or } \max_{z \in \mathcal{C}} \varphi(z) \tag{3.4}$$

The problem of CML lends itself to be solved in the framework of optimization problem. This is because, generally, any CML algorithm has to search the space of all possible maps in order to achieve maximum data association as well as to maintain minimum uncertainty in the map. Therefore, CML can be thought of as a problem of searching the most probable poses of a mobile robot for generating a 'maximally consistent' map. Consistency of a map is indicative to the quality of data association between different local maps and the amount of uncertainty associated with different obstacle points in the map. The more accurate the data association is, the lower the uncertainty is. Therefore, the term 'maximally consistent' map refers to the map having maximum data association and minimum uncertainty. The notion of 'maximal consistency' automatically imposes some constraints on the possible values of the robot poses. Additional knowledge regarding robot kinematics, dynamics and terrain type can also be applied to narrow down the search space. Therefore, the objective of CML is to maximize a map subjected to a set of constraints.

3.2.2 CML in the framework of optimization problem

A set of symbols will be specified in this section to formally define CML as an optimization problem. Some of these symbols have already been defined in Chapter 2. For further clarification, they will be re-stated (not re-defined) in this chapter.

A robot pose $\mathbf{x}_t$ is a 3-tuple $\{x, y, \theta\}$ where (x, y) is the spatial position of the robot with respect to a hypothetical coordinate system and θ is the robot's orientation. The t in subscript indicates discrete time index. The proprioceptive sensor measurement is denoted by $\mathbf{u}_t$ while $\mathbf{s}_t$ indicates exteroceptive sensor measurements. The proposed algorithm is described for a robot equipped with shaft encoder and laser range finder. Accordingly, $\mathbf{u}_t$ and $\mathbf{s}_t$ indicate odometry data and laser measurement respectively. These two sensor measurements are collected in alternation: $\{\mathbf{s}_0, \mathbf{u}_0, \mathbf{s}_1, \mathbf{u}_1, \ldots\}$. Both $\mathbf{u}_t$ and $\mathbf{s}_t$ are subjected to errors and these errors have a strong statistical dependency upon each other. A map

$\mathbf{m}_t$ generated upon receiving t-th sensor measurement $\mathbf{s}_t$ at a pose $\mathbf{x}_t$ is defined as,

$$\mathbf{m}_t = f(\mathbf{x}_{0:t}, \mathbf{s}_{0:t}) \tag{3.5}$$

where, $\mathbf{x}_{0:t} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t\}$ and $\mathbf{s}_{0:t} = \{\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_t\}$. For a single pair of robot location-sensor scan $(\mathbf{x}_t, \mathbf{s}_t)$, the functional mapping f is defined as

$$f : R(\theta_t) \cdot \mathbf{s}_t + (x_t, y_t). \tag{3.6}$$

Here, $R(\theta_t)$ is the rotation matrix defined as

$$R(\theta_t) = \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix}. \tag{3.7}$$

Equation (3.5) can be written in recursive form as

$$\begin{aligned} \mathbf{m}_t &= f(\mathbf{x}_0, \mathbf{s}_0) \cup f(\mathbf{x}_1, \mathbf{s}_1) \cup \ldots \cup f(\mathbf{x}_t, \mathbf{s}_t) \\ &= f(\mathbf{x}_{0:t-1}, \mathbf{s}_{0:t-1}) \cup f(\mathbf{x}_t, \mathbf{s}_t) \\ &= \mathbf{m}_{t-1} \cup f(\mathbf{x}_t, \mathbf{s}_t) \\ &= \mathbf{m}_{t-1} \cup \hat{m}_t. \end{aligned} \tag{3.8}$$

Here, $\hat{m}_t = f(\mathbf{x}_t, \mathbf{s}_t)$ is the local map generated from a single sensor scan according to (3.6). The robot pose $\mathbf{x}_t$ is not known unambiguously in the context of CML of mobile robot. Rather, an approximation of $\mathbf{x}_t$ can be obtained from odometry $\mathbf{u}_t$. The relation between robot pose $\mathbf{x}_t$ and the odometry $\mathbf{u}_t$ can be formalized as

$$\mathbf{x}_t = o(\mathbf{u}_t, v_t, \omega_t, d_t, F_t, K). \tag{3.9}$$

Here,

v_t = robot's linear velocity within the time interval $(t-1, t]$

ω_t = robot's angular velocity within the time interval $(t-1, t]$

d_t = total distance traveled by the robot up to time t

K = manufacturing, assembling errors and sensor resolution parameters of a robot

F_t = surface type

The quantity K in (3.9) is a vehicle specific parameter and is usually constant for a particular vehicle. To quantify F_t in (3.9), an assumption is followed that the robot is equipped with a suitable mechanism to identify different terrain conditions while traveling. Algorithms are available in literature which can extract terrain quality using sensors (e.g. vision, laser [72, 73]). However, all these quantities (v_t, ω_t, d_t, K and F_t) are related to mobile robot's odometry in a non-linear fashion. Further, they have strong dependency upon each other, and the errors they introduce in odometry accumulate unboundedly over time. Therefore, pose estimation by odometry alone results in highly erroneous map.

The non-linear mapping ϕ as described in (3.9) has not been well-defined in the literature of mobile robotics. This is because there are quantities in (3.9) whose effects on the odometry error are mostly qualitative and therefore can not be modeled in deterministic form. However, the exact calculation of $\mathbf{x}_t$ from (3.9) is complex in the context of CML. Rather, a sample based prediction $\hat{\mathbf{X}}_t$ of $\mathbf{x}_t$ can be calculated as

$$\hat{\mathbf{X}}_t = \{\mathbf{x}_t\} = \tilde{\phi}(\mathbf{u}_t, v_t, \omega_t, d_t, F_t, K) \tag{3.10}$$

where $\{\mathbf{x}_t\}$ indicates a set of robot poses which could possibly be $\mathbf{x}_t$. The function $\tilde{\phi}$ in (3.10) tries to mimic the unknown functional mapping ϕ by a subjective analysis of different quantities (v_t, ω_t, d_t, F_t and K) and their effects on the odometry $\mathbf{u}_t$. The true robot pose is searched from $\hat{\mathbf{X}}_t$ to best accommodate the sensor scan in the currently available map. Therefore, according to the definition of optimization problem, we can define the CML of mobile robot as

$$\begin{aligned} &\max \mathcal{F}(f(\mathbf{x_{0:t}}, \mathbf{s}_{0:t})) \\ &\text{s.t.} \;\; \mathbf{x_t} \in conv(\hat{\mathbf{X}}_t) \end{aligned} \tag{3.11}$$

where $\mathcal{F} : f(\mathbf{x_{0:t}}, \mathbf{s}_{0:t}) \rightarrow \Re$ is a continuous function to measure the consistency of a map. In the context of CML, there is no definite upper or lower bound for $\mathbf{x}_t$. The feasible region is defined as

$$\mathcal{C} = \{\mathbf{x} | \mathbf{x} \in conv(\hat{\mathbf{X}}_t)\} \tag{3.12}$$

Here, $conv(.)$ represents convex set. Implementation of (3.11) requires the following:

- calculation of the sample based estimate $\hat{\mathbf{X}}_t$ according to (3.10). This is related to modeling the robot's odometry while accommodating all possible sources of errors.
- an appropriate search algorithm to search for the true robot pose from $\hat{\mathbf{X}}_t$.
- defining the objective function $\mathcal{F} : f(\mathbf{x}_{0:t}, \mathbf{s}_{0:t}) \rightarrow \Xi$.

The subsequent sections of this chapter will describe the above mentioned requirements to solve CML as an optimization problem.

3.3 Fuzzy modeling of odometry error

A realistic error model of odometry must reflect the complex locomotion of a mobile robot. The most common error model described in literature is an univariate two-dimensional Gaussian. There are only a few works [29, 32, 71, 75, 76] that rigorously analyze a mobile robot's odometry . Seminal work on mobile robot's odometry described in [29] provides a comprehensive study on different properties of odometry errors. Sources of errors in the odometry fit into one of two categories, namely *Systematic errors* and *Nonsystematic errors* [29, 32]. The systematic errors typically include the following:

1. unequal wheel diameters,
2. average of wheel diameter differs from nominal diameters (due to wear),
3. finite encoder resolution,
4. finite encoder sampling rate,
5. misalignment of wheels.

Systematic errors are typically vehicle specific and do not usually change in a particular run of the robot. The quantity K defined in Section 3.2.2 of this chapter includes the sources of systematic errors. It is possible to develop deterministic mathematical model that describes the effect of K on the robot's odometry from several run of a specific vehicle. The non-systematic errors, on the other hand, are less studied in literature, though their

effects on overall odometry error is much larger than that of systematic errors in some terrains (e.g. rough outdoor environment). The sources of nonsystematic errors include the following [29, 32]

1. travel over uneven floors,
2. travel over unexpected objects on the floor,
3. wheel-slippage due to:
    - slippery floors,
    - over-acceleration,
    - fast turning (skidding),
    - external forces (interaction with external bodies),
    - nonpoint wheel contanct with the floor.

The variables that control these errors can be identified as the robot's velocity (both translational and rotational), surface type and total traveled distance (v_t, ω_t, d_t and F_t, respectively, as defined earlier in Section 3.2.2). The quantity d_t accounts for the accumulation of odometry error with time. An algorithm for multi-robot exploration [71, 75] considers both systematic and non-systematic errors to build a realistic model of odometry error. This algorithm [71, 75] devises a set of equations regarding errors in robot's translational and rotational movement after several experiments on a differential drive robot. However, the information about the dependency of a robot's odometry on the non-systematic errors are available in qualitative form, e.g. 'there will be "more" skidding if the robot turns with "high" acceleration' or 'traveling with "high" velocity on a "very" slippery floor produces "more" drift than that on a "less" slippery floor'. Because of this complex relationship, the mapping o in (3.9) lacks a deterministic mathematical structure. Instead, these qualitative information usually lead to subjective decision making about the uncertainty in robot pose. Fuzzy logic provides a natural framework to model and evaluate qualitative relations between variables [77, 78, 79]. Therefore, a fuzzy rule based model

is developed to approximate the function ϕ in (3.9) as $\tilde{\phi}$ in (3.10). The rules in the fuzzy model perform non-linear mapping from the vehicle specific constant, K, and available sensor information (about surface type F_t, velocities v_t, ω_t and traveled distance d_t) to the degree of uncertainty in odometry.

The proposed fuzzy system, termed as *Fuzzy Predictor*, has four input linguistic variables: robot's *Linear velocity* (v_t), robot's *Angular velocity* (ω_t), *Surface type* (F_t), *Traveled distance* (d_t), and five output linguistic variables: *Drift*, *Translational error*, *Rotational error*, *Vehicle specific error in orientation* and *Vehicle specific error in spatial position*. The fuzzy IF-THEN rules for each of the output fuzzy linguistic variables take the following form

- IF v_t is A_i AND ω_t is B_j AND F_t is C_k AND d_t is D_l THEN $Drift$ is E_w.
- IF v_t is A_i AND ω_t is B_j AND F_t is C_k AND d_t is D_l THEN *Translational error* is F_v.
- IF v_t is A_i AND ω_t is B_j AND F_t is C_k AND d_t is D_l THEN *Rotational error* is G_z.
- IF d_t is D_l THEN *Vehicle specific error in orientation* is H_{1q}.
- IF d_t is D_l THEN *Vehicle specific error in spatial position* is H_{2q}.

Here, $A_i, B_j, C_k, D_l, E_w, F_v, G_z, H_{1q}$ and H_{2q} are fuzzy subsets denoting linguistic values of v_t, ω_t, F_t, d_t, *Drift*, *Translational error*, *Rotational error*, *Vehicle specific error in orientation* and *Vehicle specific error in spatial position*, respectively. The linguistic labels of different fuzzy subsets are as follows

- A_i : {LOW, MEDIUM, HIGH}, $i = \{1, 2, 3\}$.
- B_j : {LOW, MEDIUM, HIGH}, $j = \{1, 2, 3\}$.
- C_k : {SLIPPERY, ROUGH }, $k = \{1, 2\}$.
- D_l, E_w, F_v, G_z : {VERY LOW, LOW, MEDIUM, HIGH, VERY HIGH }, $l, w, v, z = \{1, 2, 3, 4, 5\}$.

- H_{1q}, H_{2q} : {LOW, MEDIUM, HIGH}, $q = \{1, 2, 3\}$.

Depending on the possibility of encountering any other different surface conditions (e.g. rocky, sandy) new fuzzy subset(s) can be introduced in the model. A set of fuzzy mapping rules are formulated using the knowledge of robot's behavior subjected to different velocities, surface types and nature of accumulation of errors with time gathered from experimentations on a mobile robot and also from literatures on odometry error study [29] [71] [76]. Fuzzy subset partitions and membership function definitions are derived based on the subjective assessment of the problem. The bounds on the universe of discourse for estimating the degree of error in odometry are chosen based on experimental knowledge about the maximum possible errors in nominal condition. Membership functions for different fuzzy linguistic variables are shown in Figure 3.1 and 3.2. Table. 3.1 through 3.5 represent the rule-bases for different output variables. As an example, the rule shown in the upper left box of Table 3.1 is read as:

if Surface type is ***slippery*** *and Traveled distance is* ***Low (L)*** *and Linear velocity is* ***Low (L)****, then Drift is* ***Medium (M)****.*

The fuzzy model, termed as *Fuzzy Predictor*, is a *max-min-centroid defuzzification* Mamdani type Fuzzy Inference System (FIS) [77]. Output of the FIS is a prediction about the radius of uncertainties in spatial position (r_s) and angular orientation (r_θ) (with respect to odometry) after execution of a piece of control command.

$$\begin{aligned} r_s &= \delta s_{tr} + \delta s_K \\ r_\theta &= \delta\theta_d + \delta\theta_{rr} + \delta\theta_K \end{aligned}$$

Here, $\delta s_{tr}, \delta s_K, \delta\theta_d, \delta\theta_{rr}$ and $\delta\theta_K$ represent defuzzified values of *Translational error*, *Vehicle specific error in spatial position*, *Drift*, *Rotational error*, and *Vehicle specific error in orientation*, respectively. To account for the accumulation of odometry errors over time, accumulated uncertainty radii are calculated as

$$\begin{aligned} r_{s_t} &= r_s + r_{s_{t-1}} \\ r_{\theta_t} &= r_\theta + r_{\theta_{t-1}}. \end{aligned}$$

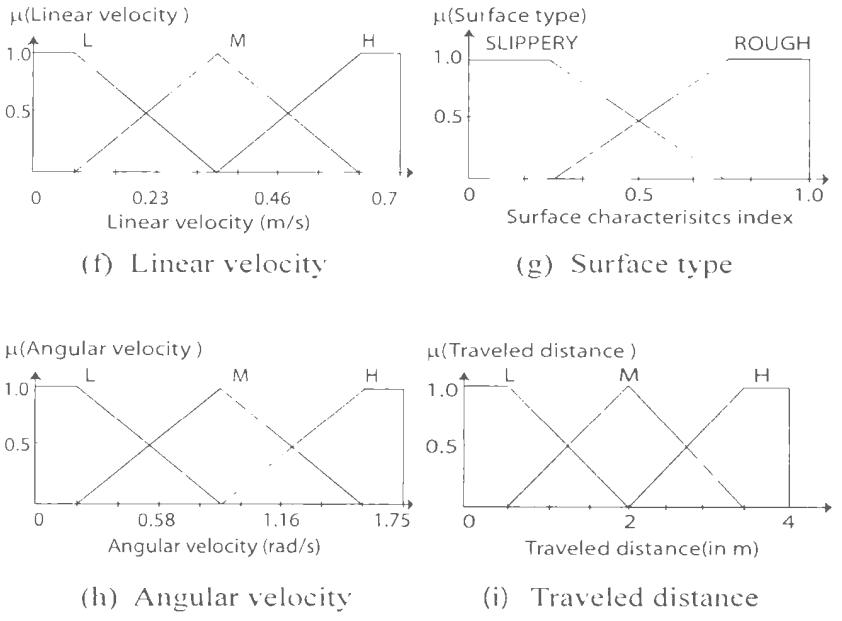


Figure 3.1: Input Membership functions

r_{s_t} and r_{θ_t} usually result in an uncertainty ellipsoid in the 3D discrete space of (x, y, θ). A number of samples are selected randomly within this uncertainty ellipsoid. These samples constitute the fuzzy sample-based prediction $\hat{\mathbf{X}}_t$ of the true robot pose $\mathbf{x}_t$. The number of samples are calculated as a fraction of the maximum possible samples within the three dimensional discrete space of predicted uncertainty

$$N_s = (N_s)_{max} c_n \tag{3.13}$$

where,

$$(N_s)_{max} = 2(\frac{2r_{s_t}}{Resolution} + 1)^2 r_{\theta_t}. \tag{3.14}$$

Here, *Resolution* indicates the resolution of a map represented in image plane in *cm/pixel* and c_n is a constant denoting a fraction of the total samples $(N_s)_{max}$. Therefore, the number of sample varies proportionately with the amount of predicted uncertainty. The dynamically varying sample size increases the probability of including the correct basin of attraction when the uncertainty in robot's pose is large, and at the same time it reduces the computational burden by generating fewer samples to take care of small uncertainty. The fuzzy sample-based prediction can be characterized by the parameter vector $[r_{s_t}, r_{\theta_t}, c_n]$.

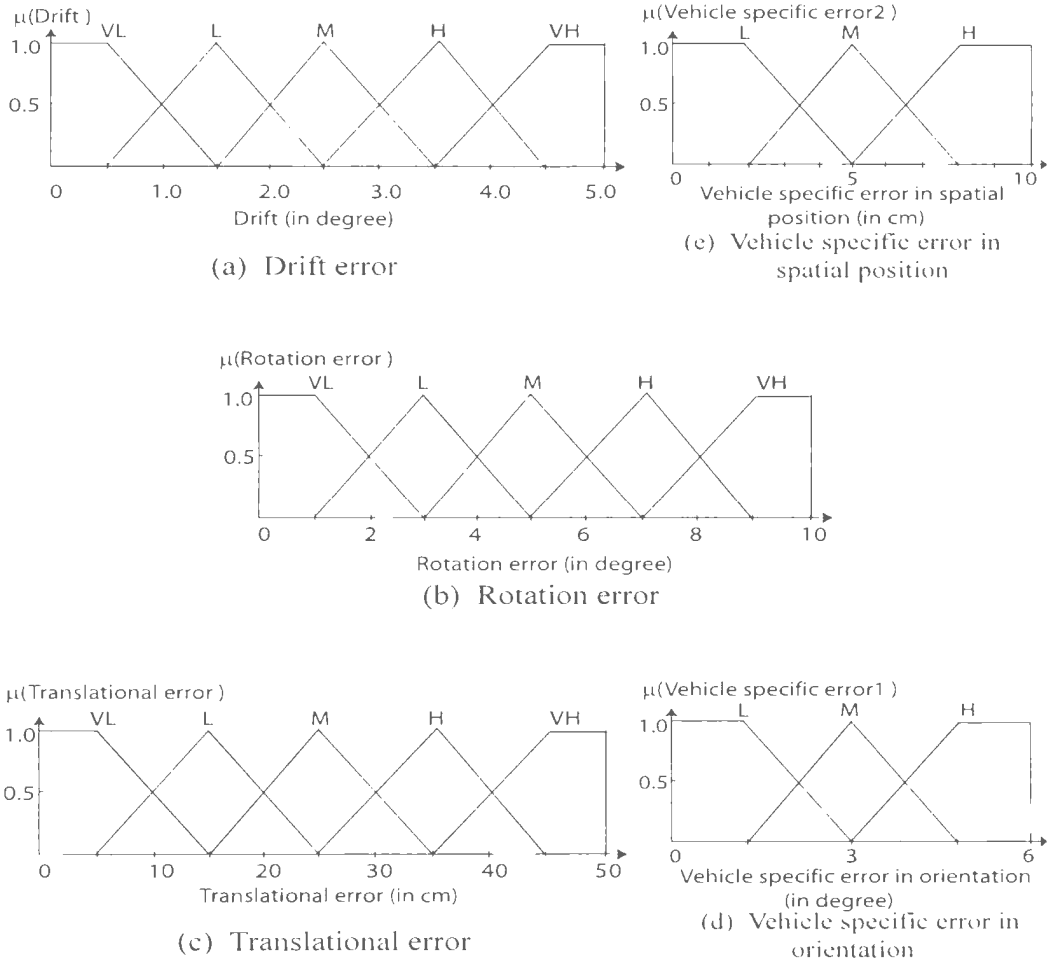


Figure 3.2: Output Membership functions

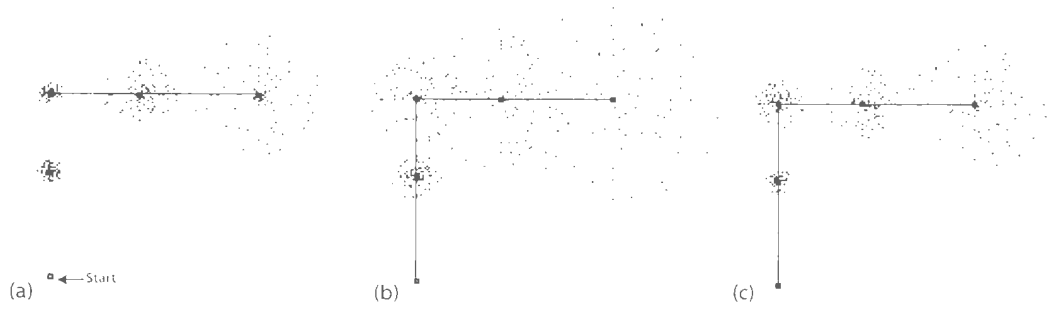


Figure 3.3: Performance of fuzzy predictor: uncertainty build up is represented by the spread of sample cloud (a) Surface: Slippery, Velocity: $0.3m/s$ (b) Surface: Rough, Velocity: $0.3m/s$ (c) Surface: Slippery, Velocity: $0.7m/s$

Figure 3.3 shows its performance in estimating uncertainty subjected to various conditions.

The control command for traversing an 'L' shaped trajectory is represented by the solid black line. The robot executes this motion command on two different type of surface, slippery (medium) and rough (medium), at two different velocities ($0.35m/s$ and $0.70m/s$). An increase in uncertainty is shown by the increasing radius of sample cloud. Figure 3.2(a) and 3.2(b) shows the effect of surface type on the amount of uncertainty. Clearly, traveling on slippery surface introduces less uncertainty than that on a rough surface (carpet) with the same velocity (this result is for a Pioneer 3 AT mobile robot whose odometry is more unreliable on carpet than on hard floor. Detail will be described in Chapter 4). For both cases, uncertainty grows with traveled distance. Similarly, the effect of robot's velocity on the amount of uncertainty is demonstrated in Figure 3.2(a) and 3.2(c) where the robot travels with different velocities on the same surface.

Table 3.1: Rule base for Drift

	Traveled distance, Linear velocity								
Surface type	L,L	L,M	L,H	M,L	M,M	M,H	H,L	H,M	H,H
Slippery	M	L	L	H	H	M	VH	VH	H
Rough	L	L	VL	H	M	L	VH	H	H

Table 3.2: Rule base for Rotation error

Surface type	*Linear velocity, Angular velocity*								
	L,L	L,M	L,H	M,L	M,M	M,H	H,L	H,M	H,H
Slippery	VL	L	M	L	M	H	L	M	VH
Rough	L	M	H	M	M	VH	M	H	VH

Table 3.3: Rule base for Translational error

Surface type	*Traveled distance, Linear velocity*								
	L,L	L,M	L,H	M,L	M,M	M,H	H,L	H,M	H,H
Slippery	VL	L	M	L	M	M	L	H	VH
Rough	L	M	M	L	M	H	M	H	VH

3.4 Search algorithm for true robot pose

A search algorithm is required to determine the true robot pose $\mathbf{x_t}$ from its sample based estimate $\hat{\mathbf{X}}_t$. The process of generating $\hat{\mathbf{X}}_t$ ensures that it is very likely that the basin of attraction of the true robot pose will be within $\hat{\mathbf{X}}_t$ or $conv(\hat{\mathbf{X}}_t)$. But any unexpected slippage or collision with obstacle may results in unusually high uncertainty and ultimately leads to the situation where $\mathbf{x}_t \notin \hat{\mathbf{X}}_t$. In such case, searching within $\hat{\mathbf{X}}_t$ or $conv(\hat{\mathbf{X}}_t)$ will detect a sub-optimal solution of the problem. Therefore, a search algorithm is required which is capable of extending the search outside $conv(\hat{\mathbf{X}}_t)$ depending on requirement.

The proposed research designs a genetic algorithm which considers $\hat{\mathbf{X}}_t$ as the initial

Table 3.4: Rule base for Vehicle specific error in orientation

Traveled distance	*Vehicle specific error in spatial position*
L	L
M	M
H	H

Table 3.5: Rule base for Vehicle specific error in spatial position

Traveled distance	*Vehicle specific error in orientation*
L	L
M	M
H	H

population. The genetic operators are designed to intelligently specify new search areas even outside of $conv(\dot{\mathbf{X}}_t)$ in order to avoid premature convergence to a local optima. The objective function $\mathcal{F}(f(\mathbf{x}_{0:t}, \mathbf{s}_{0:t}))$, as introduced in (3.2), is considered as the fitness function for the proposed GA. The functional mapping $\mathcal{F}$ is defined in such a way that $\mathcal{F}(f(\mathbf{x}_{0:t}, \mathbf{s}_{0:t}))$ faithfully measures the consistency of a map.

3.4.1 Genetic algorithm

Genetic algorithms [80] are random search techniques modeled on the principle of evolution via natural selection. They employ a population of individuals that undergo selection in the presence of variation-including operators such as mutation and crossover. A fitness (or cost) function is used to evaluate the individuals and reproductive success varies with the fitness (or cost). The string representing an individual is often referred to as its *genome*, locations on the genome are termed *loci* and the value found at a locus is an *allele*. The encoding of solution in the form of string is called a *genotype* while the solution that a specific genotype reflects is termed *phenotype*. Each iteration of a genetic algorithm, also called *generation*, involves a selection process which exhibits strong bias for fitter solutions to survive while eliminating the poorly fit solutions from the future competition. The generations continue until the termination criteria is fulfilled. A set of symbols will be specified in this section to describe the genetic algorithm designed for CML of mobile robot.

A population, denoted by $P_t(n)$, evolves from one generation, n, to the next, $n+1$. The members in a population are characterized by strings of genetic variables $\mathbf{x}_t^{\prime\prime}(n)$

(chromosome, in GA terminology). The superscript is used to label the different members of the population, where $\mu = 1, 2, 3, \ldots$.

3.4.2 Chromosome encoding: Genotype and Phenotype

The chromosomes are real-coded in the proposed GA. A chromosome represents a robot's pose. Fig. 3.4 shows the genotype of a chromosome. Its phenotype is the 3×3 homogenous transformation matrix

$$\mathbb{T} = \begin{pmatrix} R(\theta) & [x, y]^T \\ \mathbf{0} & 1 \end{pmatrix} \tag{3.15}$$

Here, $R(\theta)$ is the rotation matrix as defined in (3.7) and $\mathbf{0}$ represents a 1×2 null vector.

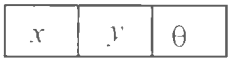


Figure 3.4: Chromosome

3.4.3 Initial population

The sample based prediction $\hat{\mathbf{X}}_t$ of the true robot pose is considered as the initial population for the proposed GA, that is $P_t(0) = \hat{\mathbf{X}}_t$. As $\hat{\mathbf{X}}_t$ infers a set of robot poses while considering possible causes of deviation from the true robot pose, it is very likely that $\hat{\mathbf{X}}_t$ will include or be near to the correct basin of attraction. This ensures fast convergence of the search process, though for $\mathbf{x}_t \notin \hat{\mathbf{X}}_t$, the partial solutions for (3.11) with relatively high fitness tend to produce clones at high rate (termed as 'founder effect' in GA terminology). This reduces the diversity in population and the GA will loose its exploratory capability and may results in a premature convergence to a local optima [81, 82]. To avoid this, a multiple island model of population [83] is adopted which helps to maintain diversity in population. The initial population $P_t(0)$ is divided into small islands $P_t(0)^m, m = 1, 2, \ldots, q$ based on the samples' spatial positions and each island evolves independently up to a predefined number of generations n_g. In addition, the evolutionary operators help to maintain the exploratory capability of the algorithm.

3.4.4 Fitness function

Design of the fitness function $\mathcal{F}$ is motivated from the scan matching approaches proposed in [41], [42] and their probabilistic generalization discussed in [31], [47]. The fitness function measures the quality of each candidate robot pose in best accommodating the current sensor scan in the partially developed global map. Therefore, the posterior $p(\mathbf{s}_t|\mathbf{m}_t, \mathbf{x}_t^\mu(n))$ is calculated as a measure of fitness for each $\mathbf{x}_t^\mu(n) \in P_t(n)$.

$$\mathcal{F}(f(\mathbf{x}_{0:t-1} \cup \mathbf{x}_t^\mu(n), \mathbf{s}_{0:t-1} \cup \mathbf{s}_t)) = p(\mathbf{s}_t|\mathbf{m}_t, \mathbf{x}_t^\mu(n)) \tag{3.16}$$

According to Bayes theorem the posterior $p(\mathbf{s}_t|\mathbf{m}_t, \mathbf{x}_t^\mu(n))$ can be expressed as

$$p(\mathbf{s}_t|\mathbf{m}_t, \mathbf{x}_t^\mu(n)) \quad = \quad \eta p(\mathbf{s}_t) p(\mathbf{m}_t|\mathbf{x}_t^\mu(n), \mathbf{s}_t) \tag{3.17}$$

where, η is the normalizer constant. The prior probability $p(\mathbf{s}_t)$ is assumed to be a narrow Gaussian centered on the zero uncertainty outcome of individual sensor measurements. This assumption performs well for high accuracy sensors [37]. The probability $p(\mathbf{m}_t|\mathbf{x}_t^\mu(n), \mathbf{s}_t)$ measures the likelihood of each obstacle point in the map $\mathbf{m}_t$ given that the current sensor scan $\mathbf{s}_t$ is taken at pose $\mathbf{x}_\mu^t(n)$. In other words, the likelihood $p(\mathbf{m}_t|\mathbf{x}_t^\mu(n), \mathbf{s}_t)$ in (3.17) measures the consistency of a map after accommodating the current sensor scan $\mathbf{s}_t$ (local map) at $\mathbf{x}_\mu^t(n)$ in the global map. Two heuristics are followed to calculate the likelihood.

1. Objects perceived by the laser range finder are opaque.

2. Space perceived as free in one sensor scan can not be perceived as occupied in the successive scan.

Similar heuristics have been used in [11], [31]. Likelihood of each obstacle point in map $\mathbf{m}_t$ is calculated based on these two heuristics. Assuming conditional independence between measurements, the resulting probabilities are multiplied to obtain the final likelihood $p(\mathbf{m}_t|\mathbf{x}_t^\mu(n), \mathbf{s}_t)$. The fitness function generates low value for poorly fit samples and higher values for better performing samples. For the real world environments, the fitness landscape of the function $\mathcal{F}$ can be fairly complex (with several local maxima or global maxima guided by valley) in the context of CML.

3.4.5 Evolutionary Operators

Selection

Selection is cross-generational and elitist [84]. Probability of an individual $\mathbf{x}_t^{\mu}(n)$ to be selected is

$$p(\mathbf{x}_t^{\mu}(n)) = \frac{\mathcal{F}^{\mu}}{\sum_{\mu} \mathcal{F}^{\mu}} \tag{3.18}$$

where $\mathcal{F}^{\mu}$ is the fitness of $\mathbf{x}_t^{\mu}(n)$. Individuals from both parent and offspring generations compete to be selected for the next generation.

Crossover

A half uniform type crossover (HUX- Half Uniform Crossover) is used. Fifty percent of the genes in a parent are crossed over with another parent. Genes to be exchanged are chosen randomly without replacement. To prevent incest, two parents having a Hamming distance of less than a certain threshold are prohibited to perform crossover. Let, $\mathbf{x}_t^{\alpha}$ and $\mathbf{x}_t^{\mu}$ are two randomly chosen parents from a population $P_t(n)$. The child through crossover, $\mathbf{x}_t^{\beta}$, is calculated as

$$\mathbf{x}_t^{\beta_i} = \chi_i \mathbf{x}_t^{\mu_i} + (1 - \chi_i)\mathbf{x}_t^{\alpha_i}. \tag{3.19}$$

and the complementary child $\mathbf{x}_t^{\bar{\beta}_i}$ is

$$\mathbf{x}_t^{\bar{\beta}_i} = (1 - \chi_i)\mathbf{x}_t^{\mu_i} - \chi_i \mathbf{x}_t^{\alpha_i}. \tag{3.20}$$

Here, χ_i is a randomly chosen binary variable. $\mathbf{x}_t^{\beta_i}$ represents i-th element of $\mathbf{x}_t^{\beta}$, $i = 1, 2, 3$. Therefore, no alleles are lost in crossover. The offsprings generated through crossover maintain diversity in population but suffer from the following limitation.

Let, the initial population contains chromosomes having x-positions $X = \{x_1, x_2, \ldots, x_M\}$, y-positions $Y = \{y_1, y_2, \ldots, y_M\}$ and orientations $\Theta = \{\theta_1, \theta_2, \ldots, \theta_N\}$, $N \leq M$. The HUX guarantees offsprings which are maximum Hamming distance away from the parents [84] but lacks the diversity that, for all offsprings, $x \in X$, $y \in Y$, and $\theta \subset \Theta$. If the actual robot pose $\mathbf{x}_t \notin \{X, Y, \Theta\}$, a premature convergence will occur. In order to avoid this situation mutation is introduced in each generation.

Mutation

Mutation plays an important role in the proposed GA to introduce new alleles (x, y or θ) in the population. Therefore, power of this GA to drive a generation towards the optimal solution, even when $\mathbf{x}_t \notin \dot{\mathbf{X}}_t$, lies partly on careful design of the mutation operator. Two mutation operators proposed by this algorithm are: *Space Mutation* and *Angle Mutation*.

Space Mutation: mutates the chromosome in such a way that the offsprings are different in spatial position while the orientation is same as the parent. Therefore, only the first two elements of chromosome undergo *Space Mutation* process.

Angle Mutation: mutates the chromosome to add diversity in orientation while keeping the spatial position unchanged. Obviously, only the third element of chromosome experiences *Angle Mutation*.

Two mutation arrays μ_S and μ_A for *Space Mutation* and *Angle Mutation*, respectively, are defined as

$$\mu_S = [a, b, 0]; a, b = [0, \pm\varepsilon]$$

$$\mu_A = [0, 0, c]; c = \pm\omega$$

The chromosome to be mutated (either *Angle Mutation* or *Space Mutation*) is added with the corresponding mutation array. The magnitudes of ε and ω determine how far the offspring will be from the ancestor in the 3 dimensional space of *(x-y-θ)*. The exploratory capability of GA is greatly influenced by the choice of these two parameters. The present implementation is performed with constant values of ε and ω. There are scopes to adjust the values of ε and ω using an intelligent algorithm such as fuzzy logic. Eligibility of a chromosome to be mutated is decided by its relative fitness in the island of population to which it belongs. Mutation shows strong bias in favor of the better performing individuals. As the selection is cross-generational and elitist, the poorly fit offsprings, if generated through mutation, never survive to the next generation.

The mutation operators enable the algorithm to generate new robot poses which might even be outside the radius of uncertainty as suggested by the *Fuzzy Predictor*. Therefore, the search space experiences a drift and the drift sustains when the true robot pose is

outside of fuzzy prediction. Crossover and mutation operators provide the GA with the capability to re-parent fresh samples from better performing ancestors. In the context of CML, this makes GA superior to other sample based methods (e.g particle filter) which update the history of a set of samples and can only perform resampling to eliminate poorly fit samples.

The tuning factors for the proposed genetic algorithm are: population size l_p, number of island m and mutation rate u_r. Together, they form the parameter vector $[l_p, m, u_r]$ that characterizes the proposed GA.

3.5 Mapping through the proposed fuzzy-GA algorithm

The proposed fuzzy-GA based algorithm processes sensor measurements sequentially. In other words, at any point in time it uses the currently available sensor readings to generate a partial map of the environment.

3.5.1 Robot pose prediction through fuzzy model of odometry

It is assumed that the robot takes a sensor scan $\mathbf{s}_0$ before executing the very first control command $\mathbf{u}_0$. Without loss of generality, the very first robot pose $\mathbf{x}_0$ is assumed to be the origin of a hypothetical global co-ordinate system and $\mathbf{s}_0$ is the available map that the robot 'memorizes' at the time of taking its first move. Let, at any point in time, $t-1$, the robot have a map $\mathbf{m}_{t-1}$ generated from currently available sensor measurements $\mathbf{s}_{0:t-1}$ and corresponding pose information $\mathbf{x}_{0:t-1}$. Execution of a new control command $\mathbf{u}_{t-1}$ makes a new sensor measurement $\mathbf{s}_t$ available. The corresponding robot pose $\mathbf{x}_t$ is the quantity to be determined. The *Fuzzy Predictor*, as described in Section 3.3, generates the sample based prediction $\dot{\mathbf{X}}_t$ to predict $\mathbf{x}_t$. The $\dot{\mathbf{X}}_t$ 'remembers' the errors in each step of the robots past moves in addition to the errors in most recent movement $\mathbf{u}_{t-1}$. Therefore, at any point in time it is very likely that the true robot pose will be included within its sample based

prediction even when the accumulated error is very high. However, this assumption does not hold good if the robot encounters slippage or collides with unexpected obstacles. In such cases *Fuzzy Predictor* usually fails to include the true robot pose within the convex set of the sample based prediction.

For the trivial case of $\mathbf{x}_t \in \dot{\mathbf{X}}_t$, the time complexity of search is $O(n)$, where n is the number of samples in $\dot{\mathbf{X}}_t$. To end up with such a trivial case it is generally required that n is very high. This makes the search almost an exhaustive one. As the robot pose is a three-dimensional continuous variable, an exhaustive search on such a huge search space does not lend the algorithm to be used online. For the non-trivial case of $\mathbf{x}_t \in conv(\dot{\mathbf{X}}_t)$, a local search is performed around each sample point in order to obtain the global maxima. An obvious choice for this is gradient ascent search. A particle filter based algorithm in [30] uses gradient ascent search starting from each particle. However, gradient ascent search imposes the constraint that $\dot{\mathbf{X}}_t$ should be densely populated with samples in order to avoid reaching local maxima. Further, gradient ascent can not guarantee global maxima for the non-trivial case of $\mathbf{x}_t \notin conv(\dot{\mathbf{X}}_t)$. A genetic algorithm is proposed in this research which performs well at all situations (i.e. $\mathbf{x}_t \in \dot{\mathbf{X}}_t$, $\mathbf{x}_t \in conv(\dot{\mathbf{X}}_t)$ or $\mathbf{x}_t \notin conv(\dot{\mathbf{X}}_t)$).

3.5.2 GA based search for true robot pose

The GA based search is designed in such a way that the convergence to global maxima does not get affected severely by the density of sample in $\dot{\mathbf{X}}_t$. Upon receiving each new sensor measurement the GA is initialized by the fuzzy sample based prediction about the robot pose. Samples in each population are divided into multiple small islands. Each population island undergoes selection, crossover and mutation processes independently up to certain number of generations, n_g. The GA-based search algorithm is described in Figure 3.5. For this implementation, $n_g = 10$. It has been observed that the GA can detect the correct basin of attraction within 10 generations. After that, individuals in different islands are merged together and the search is performed on the merged population.

After reaching global maxima, a generation starts to get populated with clones of the best estimate of true robot pose, $\tilde{\mathbf{x}}_t$. Repetition of the same maximum fitness for gener-

Algorithm GA_Search($\hat{\mathbf{X}}_t, \mathbf{s}_t$):

$count$=0; // a variable to count the number of generations.

$P_t(count) := \hat{\mathbf{X}}_t$ //fuzzy prediction is set as initial population.

while($count < n_g$) //continues upto n_g generations.

$P_t(count) = \bigcup_{a=1}^{4} P_t^a(count)$ // population is divided into small islands.

for $a = 1, 2, 3, 4, \ldots$ // for each island of population

Fitness evaluation according to (3.17)

$P_t^a(count) \xrightarrow{selection} [P_t^a(count)]^{sel}$ // selection according to (3.18)

$[P_t^a(count)]^{sel} \xrightarrow{mutation} [P_t^a(count)]^{sel,mut}$ //mutation

$P_t^a(count) \xrightarrow{crossover} [P_t^a(count)]^c$ //crossover according to (3.19-3.20)

Fitness evaluation according to (3.17)

$P_t^a(count) \cup [P_t^m(count)]^c \cup [P_t^a(count)]^{sel,mut} \xrightarrow{selection} P_t^a(count+1)$

end for

$count++$

end while

Figure 3.5: the GA-based search Algorithm

ations and monotonically decreasing diversity of population is considered as the stopping criteria for the proposed GA. Despite the best estimate $\tilde{\mathbf{x}}_t$, the associated terminating generation is also 'remembered' with each robot pose in order to perform the back propagation of accumulated error.

3.5.3 Map update and backward correction of map for loop closing

After obtaining the best estimate of robot pose, the map is updated according to (3.5) - (3.8). As long as the robot explores new area from each new scan, the accumulated error in pose estimation does not deteriorate the topological consistency of the map. However, whenever any previously mapped area is discovered again from the current sensor scan (e.g. during closing a loop in a cyclic environment), a severe topological inconsistency appears

in the map. The amount of inconsistency depends on the dimension of the loop and also on the type of environment. From experimentations in different types of cyclic environments it is observed that the loop closure error can be extremely high, specially during mapping long featureless corridors where there are insufficient features for localization and the end points (of the corridor) are not detectable reliably from the other end. Backward correction of the calculated robot poses is usually proposed in literature [13], [59] to take care of this problem. The loop closure process in this research essentially follows the similar approach of backward correction. A directed graph of the calculated robot poses is always maintained in order to detect the loop closure. A node in this graph indicates the spatial position of a robot while an edge represents the difference in orientation between two successive nodes. For example, at time $t = 1$, if the calculated robot poses according to the proposed algorithm are $[x_1, y_1, \theta_1]$, $[x_2, y_2, \theta_2]$, $[x_3, y_3, \theta_3]$ and $[x_4, y_4, \theta_4]$, then the directed graph $G = \{V, E\}$ is defined as

$$\begin{aligned} V &= \{\{x_1, y_1\}, \{x_2, y_2\}, \{x_3, y_3\}, \{x_4, y_4\}\} \\ E &= \{\alpha_1, \alpha_2, \alpha_3\} \end{aligned} \tag{3.21}$$

where,

$$\begin{aligned} \alpha_1 &= \theta_2 - \theta 1 \\ \alpha_2 &= \theta_3 - \theta 2 \\ \alpha_3 &= \theta_4 - \theta 3 \end{aligned}$$

In a two-dimensional Cartesian coordinates system (same as the global coordinate system of the map) the graph looks like Figure 3.6. To detect a loop closure two simple heuristics are applied.

1. $250^o < \sum_{t-1} \alpha_{t-1} \leq 300^o$
2. $\exists_{m \in \{1, \ldots t\}} \cdot d(\{x_{t+}, y_{t+}\}, \{x_m, y_m\})$ is decreasing monotonically. Here, $d(.)$ represents Euclidean distance between two points, and $t+$ in the subscript indicates time in near future including the present time.

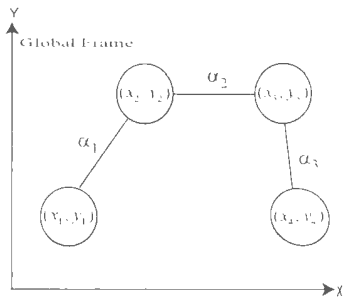


Figure 3.6: Directed graph for loop closure detection.

For any calculated robot pose $\mathbf{x}_t$, if the first condition is satisfied, it is assumed that the corresponding pose is 'possibly' going to close a loop. If the successively calculated robot poses satisfy the second condition, it is decided that $\mathbf{x}_t$ and its successors 'certainly' close a loop. Whenever a loop closure is detected, a backward correction of poses is performed starting from $\mathbf{x}_t$ up to the neighborhood of $\mathbf{x}_m$. A genetic algorithm similar to the one described in Section 3.4 takes care of the backward correction process. For every pose $\mathbf{x} \in [\mathbf{x}_t, \mathbf{x}_m]$, a GA searches for the corrected pose which maximize the consistency of loop. Here, search starts from the terminating generation which is 'remembered' along with the best estimate of the true robot pose.

Strictly speaking, the procedure of loop closure detection might not work well for a complex loop structure or a nested loop.

3.6 Summary

The proposed CML algorithm, presented in this chapter, is an incremental mapping algorithm which fulfills all necessary requirements of a robotic mapping algorithm. The algorithm uses fuzzy logic to model the error in a mobile robot's odometry. The fuzzy modeling of error in odometry enables the algorithm to consider all the systematic and non-systematic errors while inferring the uncertainty in robot pose. Compared to most of the existing CML algorithms, the proposed algorithm does not confine itself with a Gaussian assumption of odometry error. The fuzzy error model of odometry generates a sample based prediction of the robot pose. A GA is designed to search for the optimal

robot pose which best accommodate a local map in the current partially developed global map. This GA based search starts from the pose space defined by the fuzzy system. The property of *natural selection*, which supports the survival of better performing individuals, offers an iterative solution to the correspondence problem of robotic mapping. The capacity of GA to perform both resampling from the existing samples and at the same time re-parenting fresh samples imposes less restriction on the initial sample distribution (provided by the fuzzy error model of odometry) to closely resemble the original distribution. This also enables the algorithm to avoid converging to a local optima. Experimental results are described in the next chapter to validate the performance of the proposed algorithm.

Chapter 4

Results

About this chapter. This chapter describes experimental results to validate the proposed robotic mapping algorithm. A small discussion on the equipments used to carry out the experiments are provided first. This follows the description and analysis of experimental results obtained from various simulated and real world environments.

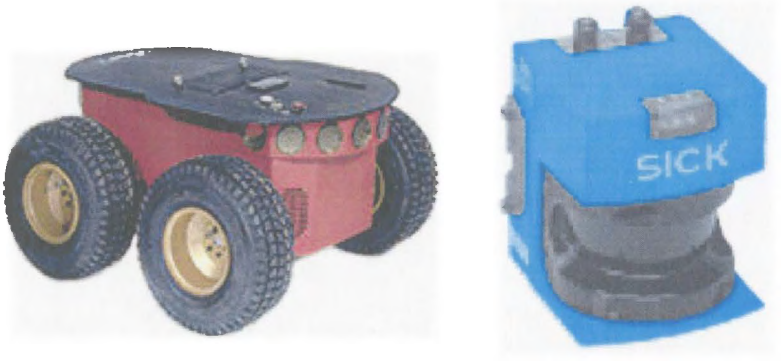
4.1 Robot and sensor system

An ActivMedia Pioneer 3 AT mobile robot is used in this research to test the proposed algorithm. A SICK LMS 200 laser range finder is used as an exteroceptive sensor and the optical encoder of the P3 AT is used as a proprioceptive sensor.

4.1.1 ActivMedia Pioneer 3 mobile robot

The ActivMedia Pioneer 3 AT mobile robot (Figure 4.1(a)) used in this research is a 4 wheel drive, all-terrain mobile robot which operates autonomously with an onboard PC104 computer and multiple PC104 accessory cards. The drive system of this robot uses two high-speed, high-torque reversible DC motors, each equipped with a high-resolution optical quadrature shaft encoder for precise position and speed sensing and advanced dead-reckoning. The robot tracks its position and orientation based on dead-reckoning

from wheel motion derived from encoder readings (motor encoder resolution is 500-ticks per revolution). The Active Media Robot Control and Operation Software (ARCOS) maintains robot's internal coordinate position in platform-dependent units, but reports the values in platform-independent millimeters and degrees in the standard Server Information Packet (SIP). This registration between external and internal coordinates deteriorates rapidly with movement due to gearbox play, wheel imbalance and slippage, and many other real-world factors. The dead-reckoning ability of the robot is reliable for a short travel distance; in the order of a few meters, or one to two wheel revolutions, depending on the surface. Traveling on carpet produces more erroneous result than that on the hard floors [85]. Also, moving either too fast or too slow tends to exacerbate the absolute position errors. Accordingly, the robot's dead-reckoning capability can be considered as a means of tying together sensor readings taken over a short period of time, not as a method of keeping the robot on course with respect to a global map. The maximum linear velocity of P3 AT on flat terrain is 0.7m/s.



(a) ActivMedia Pioneer 3 AT mobile robot

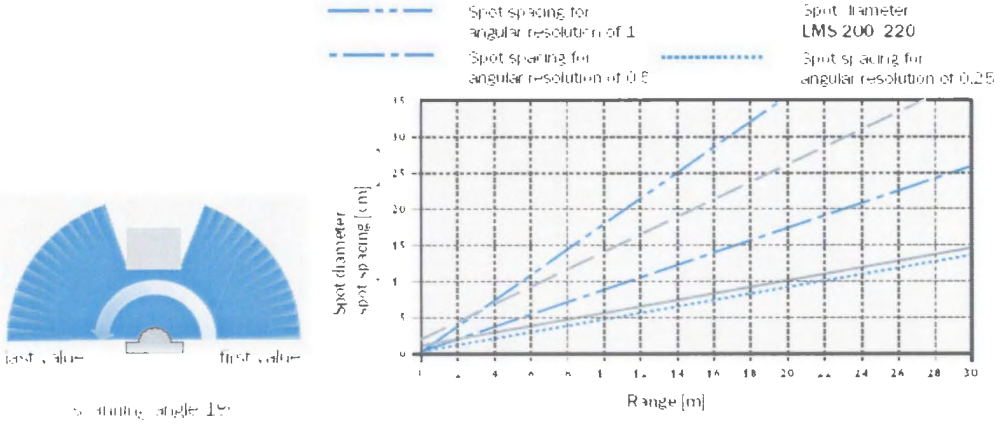
(b) SICK LMS 200 Laser range finder

Figure 4.1: Robot and sensor used for experiment

4.1.2 SICK LMS 200 laser range finder

The Laser Measurement System LMS 200 laser range finder developed by SICK is a non-contact measurement system that scans the surroundings two-dimensionally. As a scanning

system, the device requires neither reflectors nor position marks. The LMS 200 system operates by measuring the time of flight of laser light pulses: a pulsed laser beam is emitted and reflected if it meets an object. The reflection is registered by the scanner's receiver. The time between transmission and reception of the impulse is directly proportional to the distance between the scanner and the object (time of flight). The pulsed laser beam is deflected by an internal rotating mirror so that a fan-shaped scan is made of the surrounding area (laser radar) (Figure 4.2(a)) The maximum scanning angle is 100^o or 180^o (180^o scanning angle is used in this research). The contour of the target object is determined from the sequence of impulses received. The measurement data is available in real time for further evaluation via a serial interface. Two important parameters of LMS system are the variation of spot spacing and spot diameter with range. In a radial field of vision, a light impulse (spot) is emitted every $0.25^o, 0.5^o$ or 1^o (0.5^o resolution is used in this research). As a result of the beam geometry and the diameter of the individual spots, the spots overlap on the target object up to a certain distance. Figure 4.2(b) shows spot spacing in relation to the range and the corresponding spot diameter for the LMS 200. The range of the scanner depends on the reflectivity of the target object and the

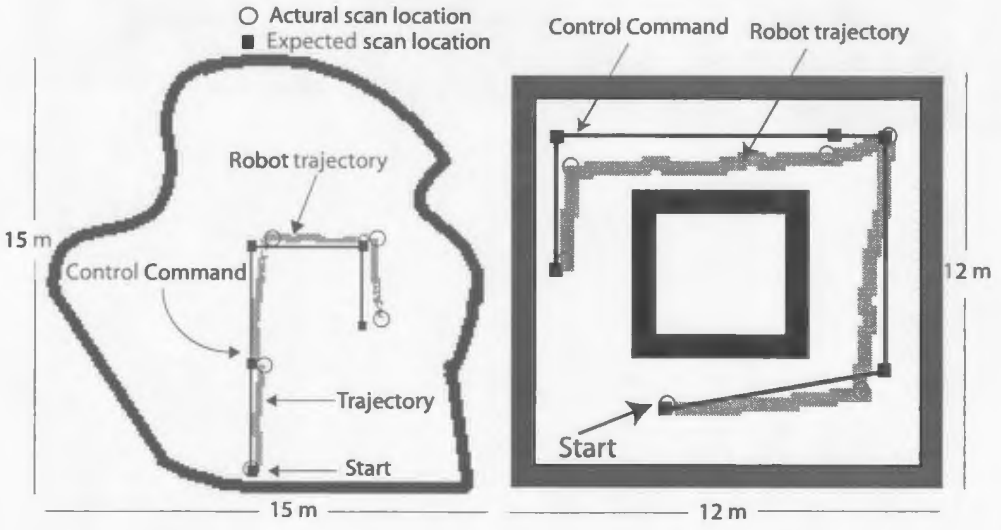


(a) Fan shaped scanning of LMS 200

(b) Variation of spot size and spot spacing with range

Figure 4.2: LMS 200 Operating characteristics [Adopted from [86]]

transmission strength of the scanner. In standard setting, the maximum range is 30 meter



(a) Polygonal environment (b) Cyclic environment

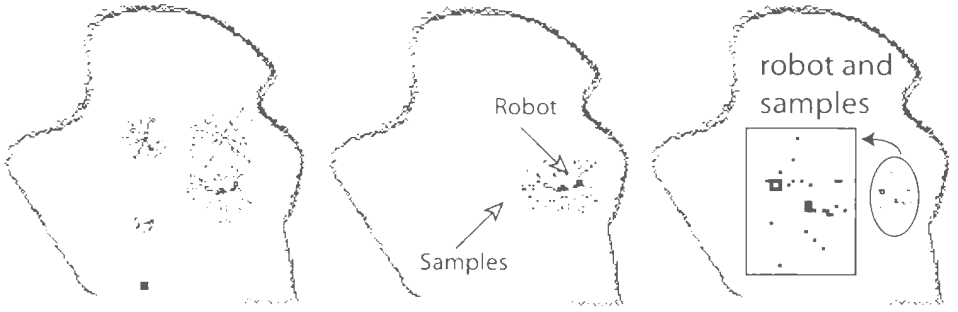
Figure 4.3: Simulated indoor environments

with 10% reflectivity [86].

4.2 Simulation result

The proposed algorithm is first tested on synthetic data of different indoor environments using a simulated robot and laser range finder. The simulated robot has maximum speed of 0.7m/s. The simulated laser range finder has maximum range of 30m with an angular resolution of 1^o for maximum scanning angle of 180^o. Environments are modeled using straight lines and spline curves. The robot is instructed to take scan at some pre-specified locations. Random noise is introduced in its motion (both in translation and orientation). The added orientation error varies from -15^o to 15^o while the translational error is randomly distributed within a circle of radius $\pm$30.48cm. Measurement noise are modeled as random variable uniformly distributed within a circle of radius $\pm$15.24cm.

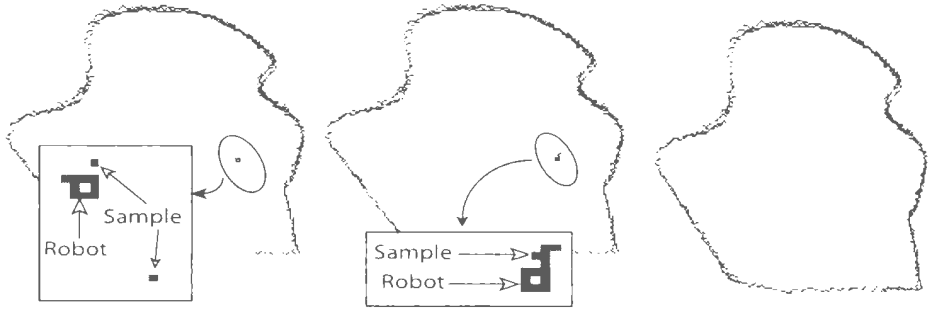
The first simulated test environment is a 15m $\times$15m polygon consist of lines and splines (Figure 4.3(a)). The pre-specified scan locations are marked by solid black boxes and the circles show the locations where the robot actually takes scan due to errors in executing



(a) Predicted uncertainty at different scan location

(b) Initial population at scan location five

(c) Generation 1 at scan location five



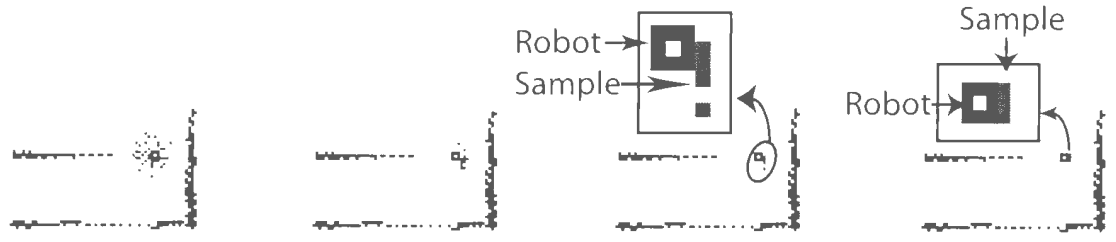
(d) Generation 2

(e) Generation 3

(f) Final map

Figure 4.4: Mapping a polygonal environment

motion commands. Uncertainty in the robot pose at different scan locations are shown in Figure 1.4(a). Growth of uncertainty is represented by increasing radius of sample cloud around each expected robot location. Figure 4.1(b) shows a scenario when the robot is at the last scan location and has already developed a partial map of the environment. The performance of GA module in searching the actual robot pose is shown in Figure 1.4(c)-1.1(e). The samples in Figure 1.4(b) constitute the initial population. The samples start to generate around the true robot pose (Figure 1.1(c)-1.1(e)) as the generations evolve. The algorithm converges after four generations. The initial population contains 204 samples. When the robot disambiguates its position uncertainty, the new sensor measurement is added at the calculated robot pose (Figure 1.1(f)).

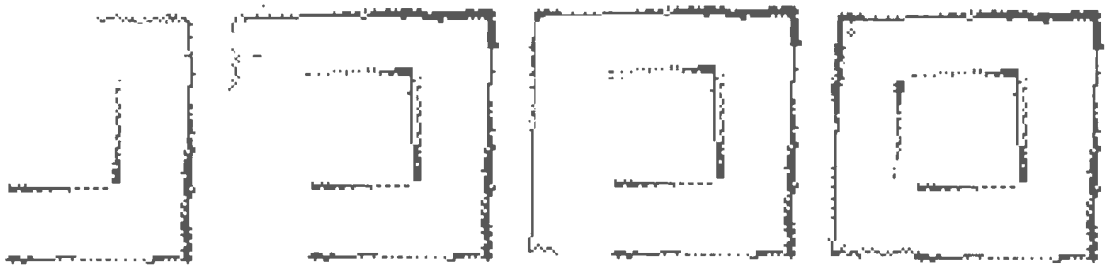


(a) Initial population at second scan location

(b) Generation 1

(c) Generation 2

(d) Generation 3



(e) Map generated after adding second scan

(f) after adding fourth scan

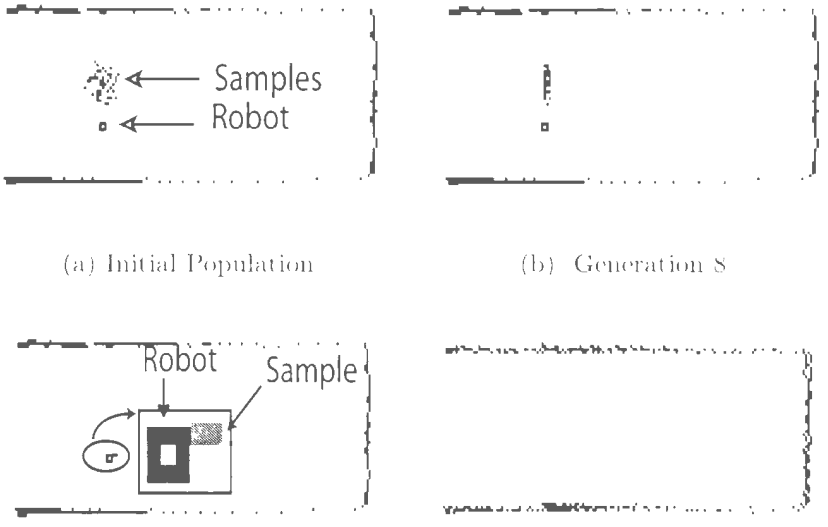
(g) after adding fifth scan

(h) Final map

Figure 4.5: Mapping a simulated cyclic environment

Another simulated test environment is shown in Figure 4.3(b). It is cyclic hallway of size 12m ×12m. The robot is instructed to capture six scans at six different locations (Figure 4.3(b)). Figure 4.5(a)-4.5(h) show different stages of the incremental mapping process. In Figure 4.5(a), the robot is at its second scan location with the partial map of environment captured at the starting pose. It disambiguates the position uncertainty gradually and estimate a pose which is close to the actual pose. Similar analysis is applicable to the other scan locations. The final map obtained (Figure 4.5(h)) is not perfectly accurate as the walls are sometimes not properly aligned. However, the final map is appropriate for navigation and does not contain any topological inconsistency.

The initial sample space, in the above examples, generated by the *Fuzzy Predictor*



(a) Initial Population (b) Generation 8

(c) Generation 17 (d) Final Map

Figure 4.6: Robustness of Genetic algorithm

includes the actual robot pose. The proposed GA is capable to generate poses near the actual robot pose if the *FuzzyPredictor* does not include it. Figure 4.6(a) shows such a situation. The robot's actual position is outside the predicted search space. Figure 4.6(b) to 4.6(c) show the gradual drift of the sample space toward the actual solution. The algorithm converges after seventeen generations. The resulting map is shown in Figure 4.6(d).

4.3 Experimental result

The proposed algorithm is tested on real world data captured by ActivMedia Pioneer 3 AT mobile robot equipped with SICK LMS 200 laser range finder (experimental set up is shown in Figure 4.7). For mapping, the robot is operated by a joystick to different places at Memorial University of Newfoundland. The robot is instructed to capture sensor reading after each 1 to 1.5 meter of travel or 30° to 40° change in orientation. Gyroscope correction of the robot is turned off to obtain raw odometry reading. The Pioneer 3 AT can not turn without skidding due to lack of differential drive mechanism. This fact

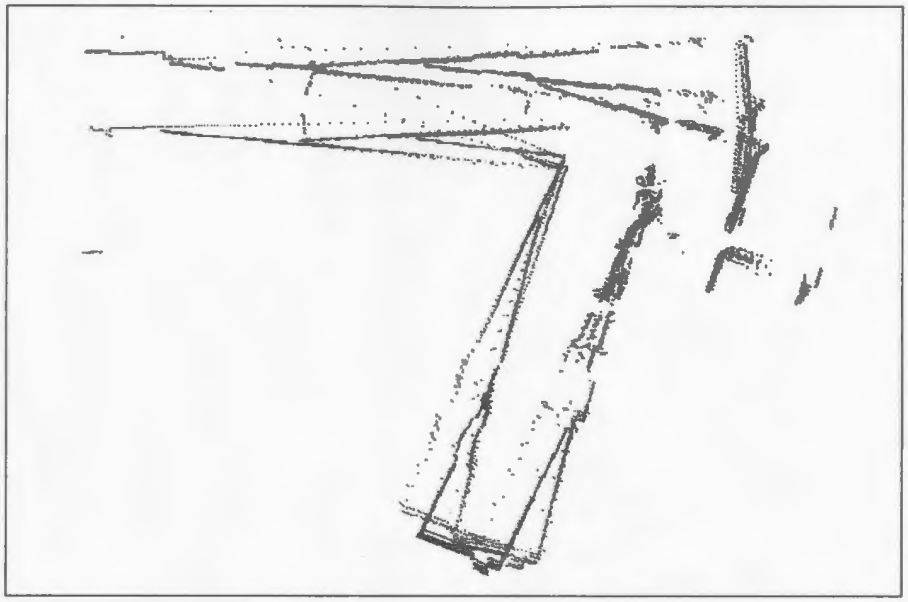
Figure 4.7: Robot and sensor setup for experiment

supports our experimental observation that the translational error is larger compared to rotation error when the robot undergoes frequent turning. The fuzzy sets as well as the rule bases are constructed after observing the behavior of odometry errors from several runs of the robot on two surface conditions (tile and carpet). As discussed in Section 4.1.1, the odometry readings are more reliable while traveling on tile than those on carpet. In order to justify the capability of the algorithm to handle erroneous odometry, the robot is pushed often by hand to introduce more errors in odometry. The added orientation error at each 'manual push' varies from $0^o - 40^o$ and the translation error from $0 - 80$cm.

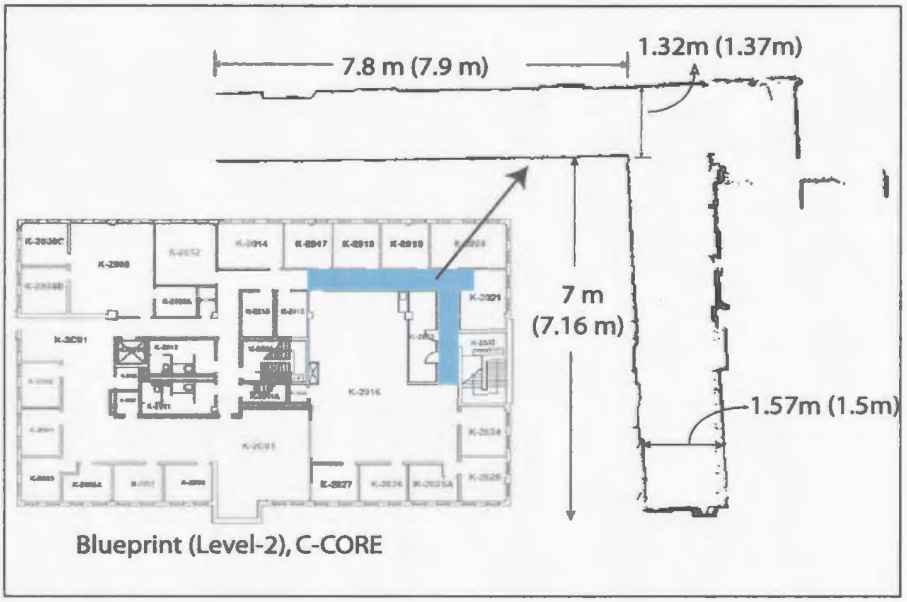
4.3.1 Test 1:

- **Environment:** $15m$ long L-shaped corridor (blueprint shown in Figure 4.8(a))
- **Surface:** Tile floor
- **Range measurement:** After each 1.5 meter of travel or 40^o change in orientation
- **Added odometry error:** Extremely high

Due to high odometry error, in most cases the *Fuzzy Predictor* fails to include the true robot pose within the sample based prediction. The map generated from raw odometry



(a) Map from odometry data. total 21 scans



(b) Final map generated after fuzzy based prediction and GA based global search. Blue shaded area in the blueprint shows the same corridor

Figure 4.8: Mapping an approximately $15m$ long corridor under severe odometry error introduced by 'manually pushing' the robot. Quantities within bracket show actual dimensions. Other dimensions are according to the developed map

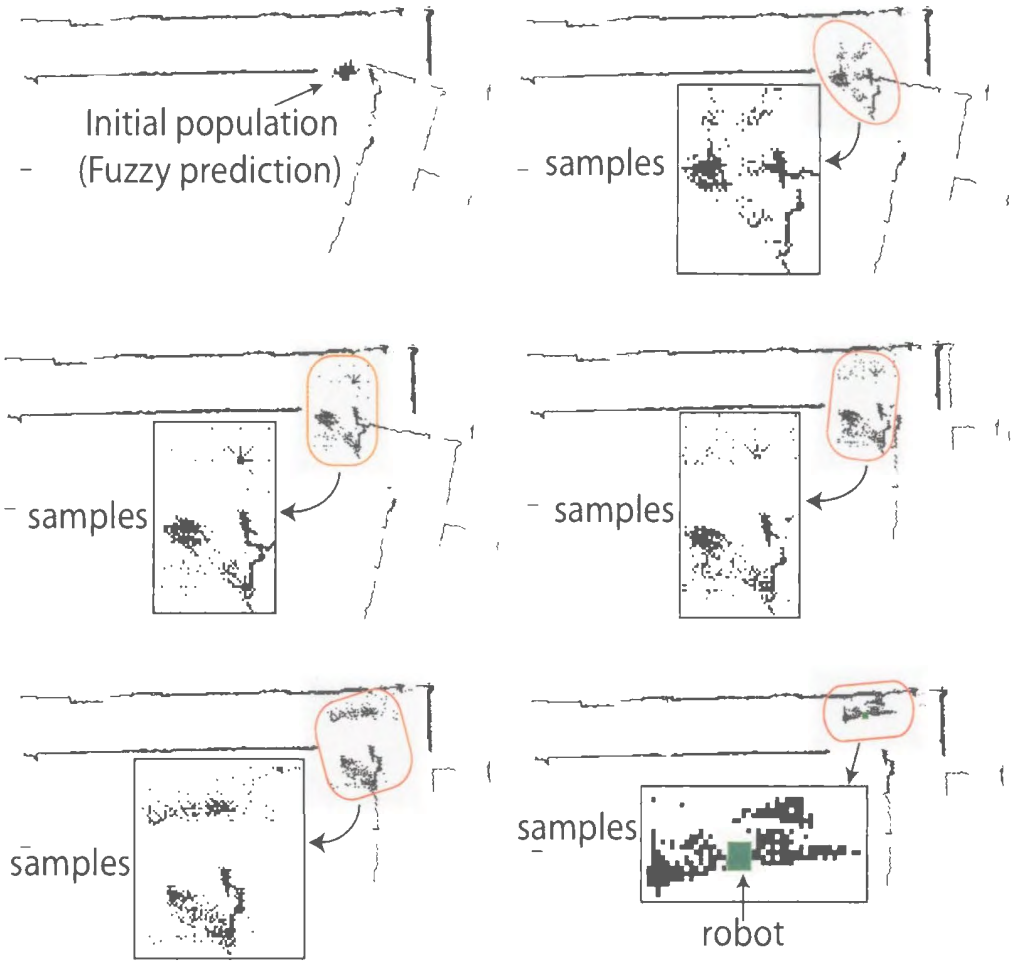


Figure 4.9: Performance of genetic algorithm in registering a local map (scan no. 15) during mapping the environment shown in Figure 4.8 (a) Initial registration by odometry. Fuzzy sample based prediction for robot pose (initial population for GA) is indicated by the point cloud (b) Local map registration according to the highest fit member of first generation. Samples are generating outside initial population (c) Second generation. More than one island of populations are evolving independently (d) Third generation. Samples have generated within the correct basin of attraction (e) Fifth generation. The best estimate of true robot pose has determined (f) Terminating generation. Samples are gathered around the best estimate of true robot pose

data is shown in Figure 4.8(a). This map contains severe topological inconsistency and does not reflect the true geometry of the building. The odometry readings are corrected according to the proposed algorithm and this results in the map as shown in Figure 4.8(b) after registering 21 scans. Figure 4.9(a)-4.9(f) show different generations of the genetic algorithm while accommodating a new local map (scan number 15) in the partial global map developed from previous sensor measurements. This particular local map is captured after pushing the robot approximately 1m away from its true position and turning it manually by an angle of approximately 21°. Parameter vector for the *Fuzzy predictor* is $12cm, 5°, 20$]. Two-dimensional projection of the samples on the x-y plane are shown by the black dots. Each image in Figure 4.9 shows the alignment of the local map according to the highest fit chromosome of a generation. Whenever the GA locates the best estimate of the true robot pose, the samples start to generate around the best estimate and the population rapidly starts to lose diversity.

4.3.2 Test 2:

- **Environment:** Cyclic environment of approximately $17m \times 11m$ dimension (blueprint shown in Figure 4.10).
- **Surface:** Tile floor and carpet.
- **Range measurement:** After each $1m$ of travel or $20°$ change in orientation.
- **Added odometry error:** Moderate.

As discussed in Section 4.1.1, traveling on carpet introduces more error in odometry. The trajectory is chosen in such a way that the robot is subjected to frequent turning. Therefore, the resulting map from raw odometry data (Figure 4.11) shows large translational errors. This environment is not completely corridor-type and contains several features (e.g. table, chair, door etc.). Therefore, accumulated errors during loop closure do not become too high. The corrected map as shown in Figure 4.12 consists of 45 local maps

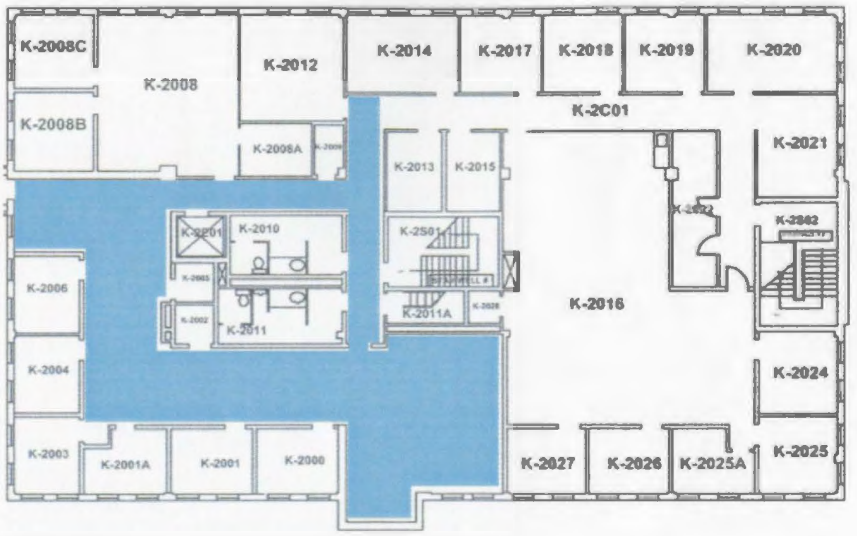


Figure 4.10: Architectural blueprint of Level-2, C-CORE (courtesy of C-CORE, MUN). The blue shaded area shows a $17m \times 14m$ loop mapped by the robot

and maintains topological consistency (the generated map does not exactly reflect the blue print because of tables, chairs and temporary partitions in the environment).

4.3.3 Test 3:

- **Environment:** Larger cyclic hall-way of approximately $22m \times 12m$ (blueprint is shown in Figure 4.13).
- **Surface:** Tile floor.
- **Range measurement:** After each $1m$ of travel or 20^o change in orientation.
- **Added odometry error:** None.

This environment is more challenging than the cyclic environment of Test 2 because, the hall-way does not contain much detectable features to facilitate robot localization. Moreover, the end points of the hall-way are not measurable reliably from the other end. Therefore, pose errors accumulate while adding local maps and become very high during loop closing. Map generated using raw odometry data is shown in Figure 4.14. The red

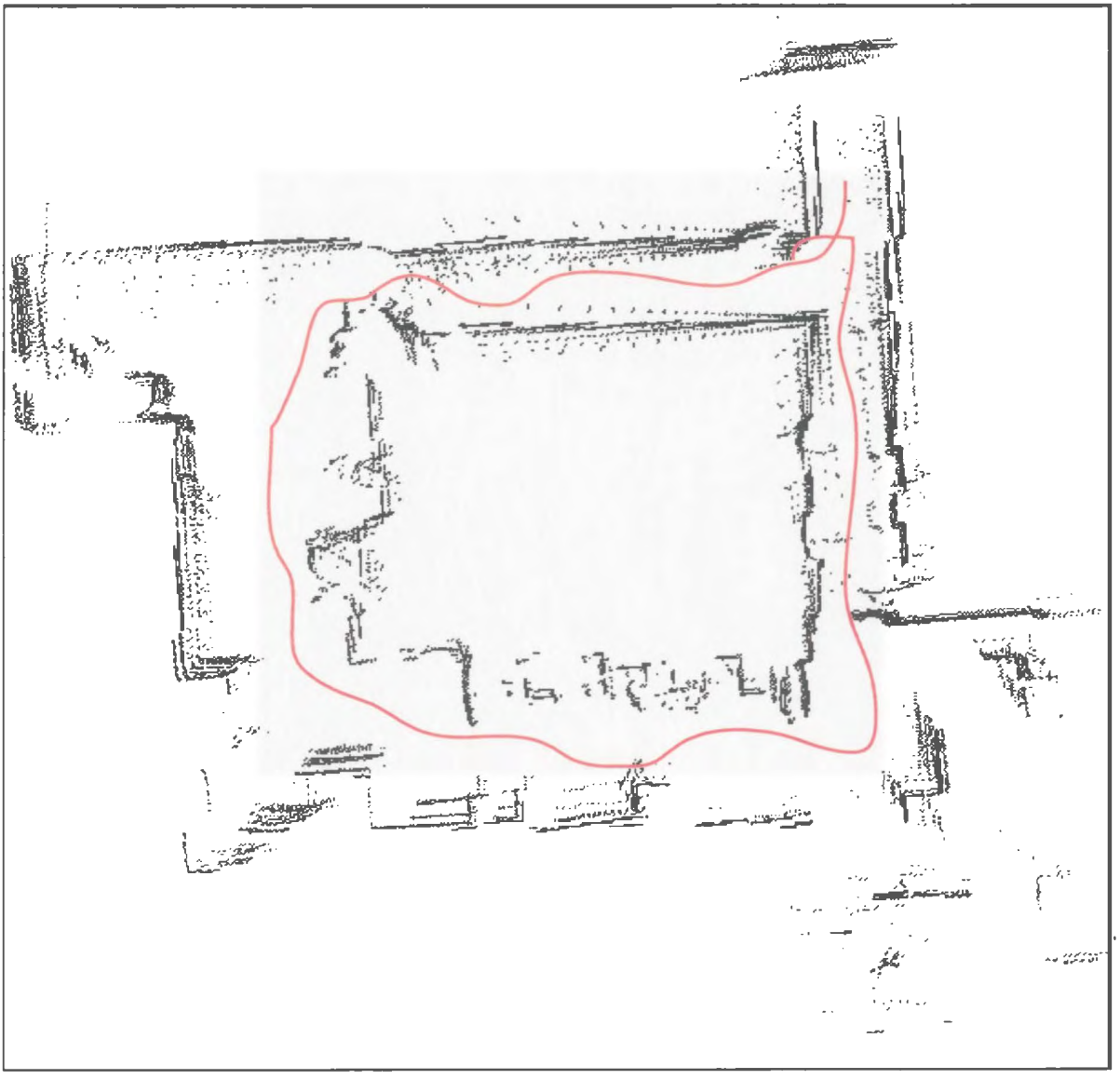


Figure 4.11: Mapping cyclic environment of dimension $17m \times 14m$: Map from raw odometry data. The red line shows robot trajectory according to odometry

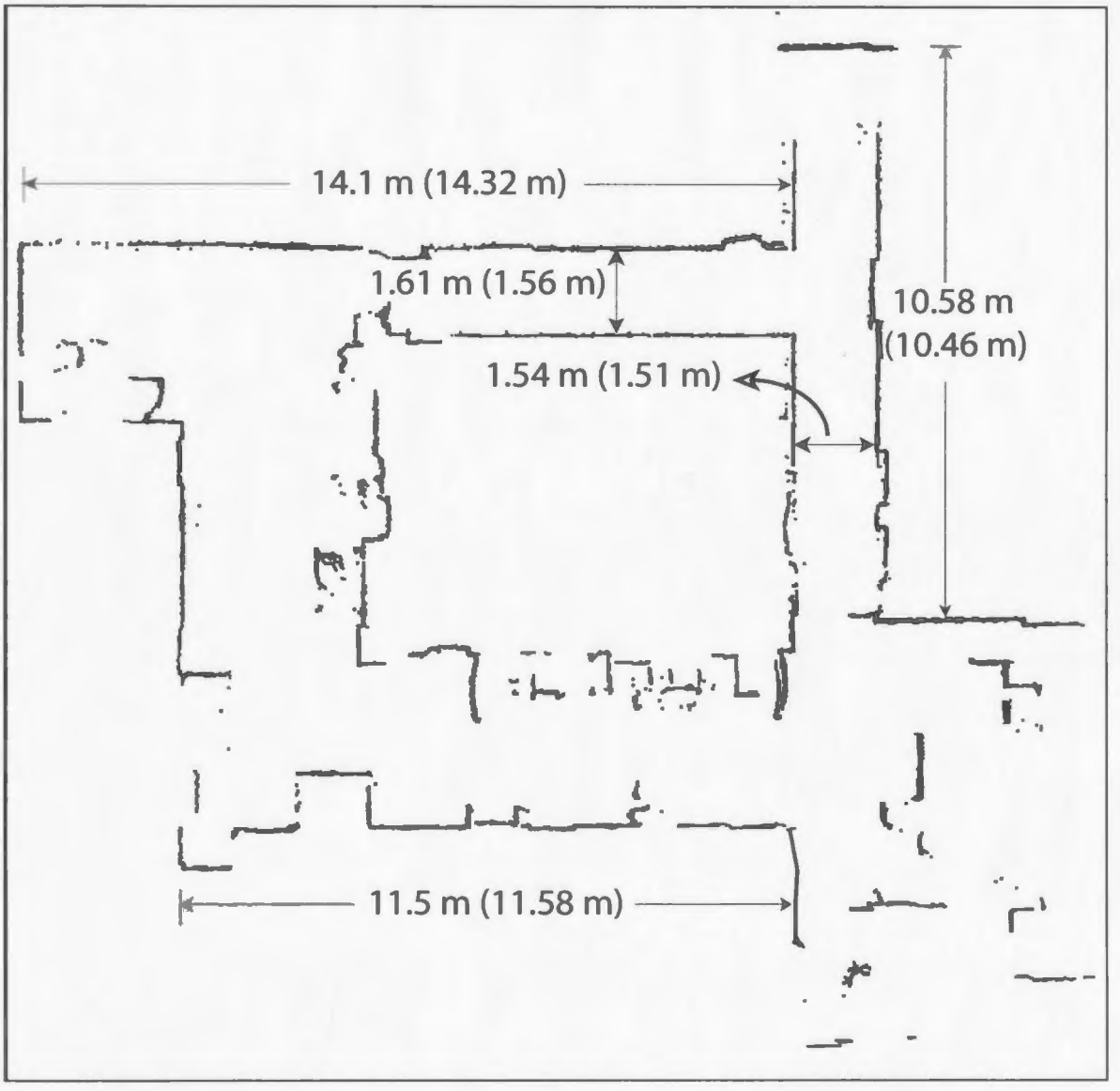


Figure 4.12: Mapping cyclic environment of dimension $17m \times 14m$: Map generated using proposed CML algorithm. Quantities within bracket show actual dimensions. Other dimensions are according to the developed map

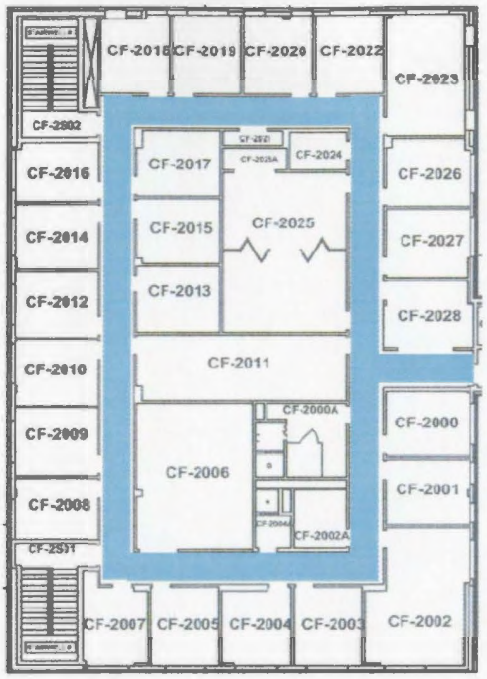


Figure 4.13: Blueprint of Level-2, Centrifuge building. Blue shaded area shows a $22m \times 12m$ hall-way mapped by robot

line shows the odometry. The back propagation of accumulated errors help to recover the final map as shown in Figure 4.15. This map consists of 115 local maps and reflects the correct geometry of the actual environment with minimum uncertainty.

4.3.4 Test 4:

- **Environment:** $17m \times 15m$ Intelligent System Lab(ISLAB), C-CORE (blue print shown in Figure 4.16)
- **Surface:** Tile floor.
- **Range measurement:** After each $1m$ of travel or 30^o change in orientation.
- **Added odometry error:** None.

The interior of ISLAB is highly unstructured unlike other environments described earlier in this section. The map generated from raw odometry data is shown in Figure 4.17. The

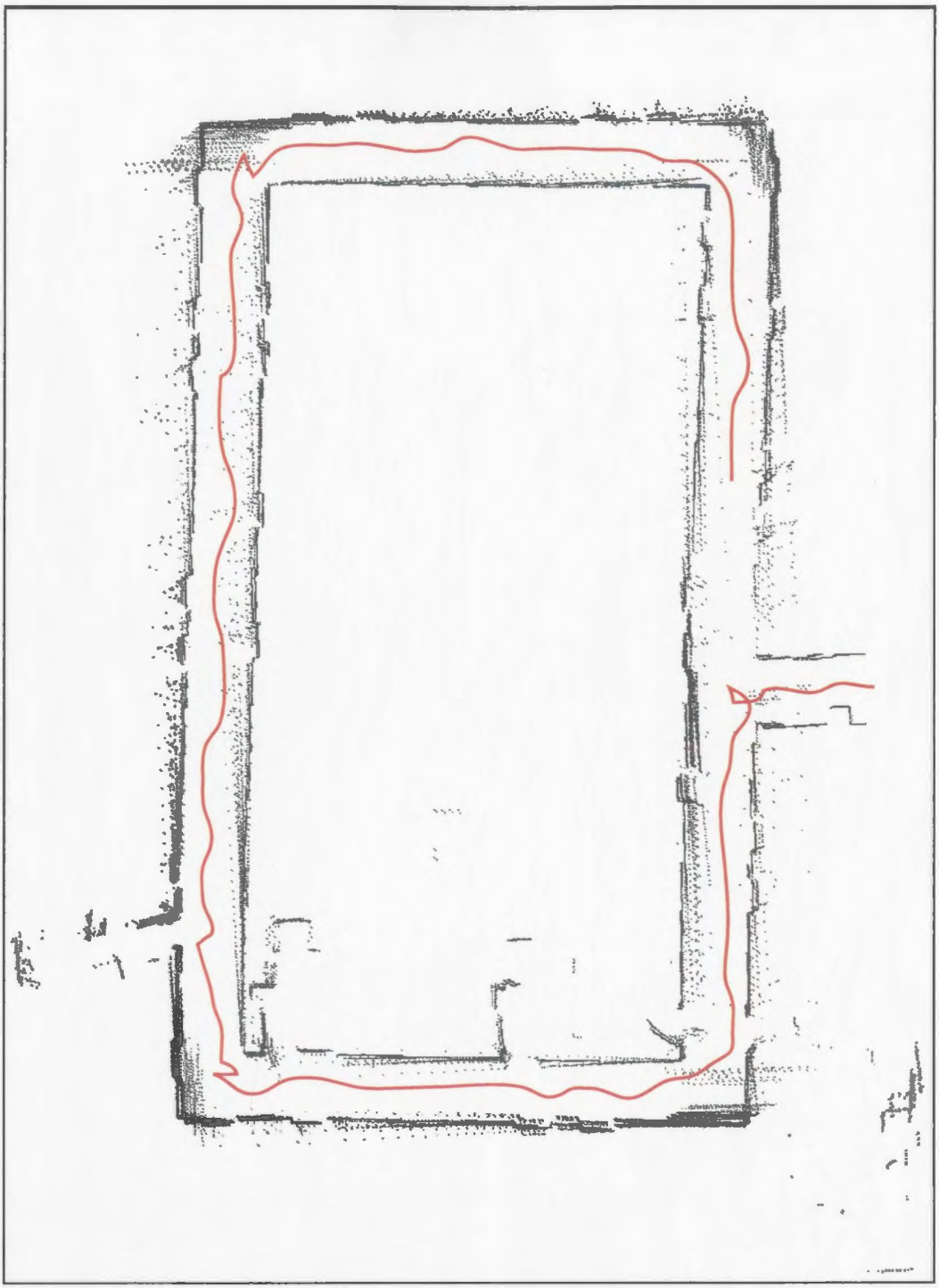


Figure 4.14: Mapping a large cyclic corridor of dimension $22m \times 12m$: Map from odometry. Red line shows the robot trajectory according to odometry

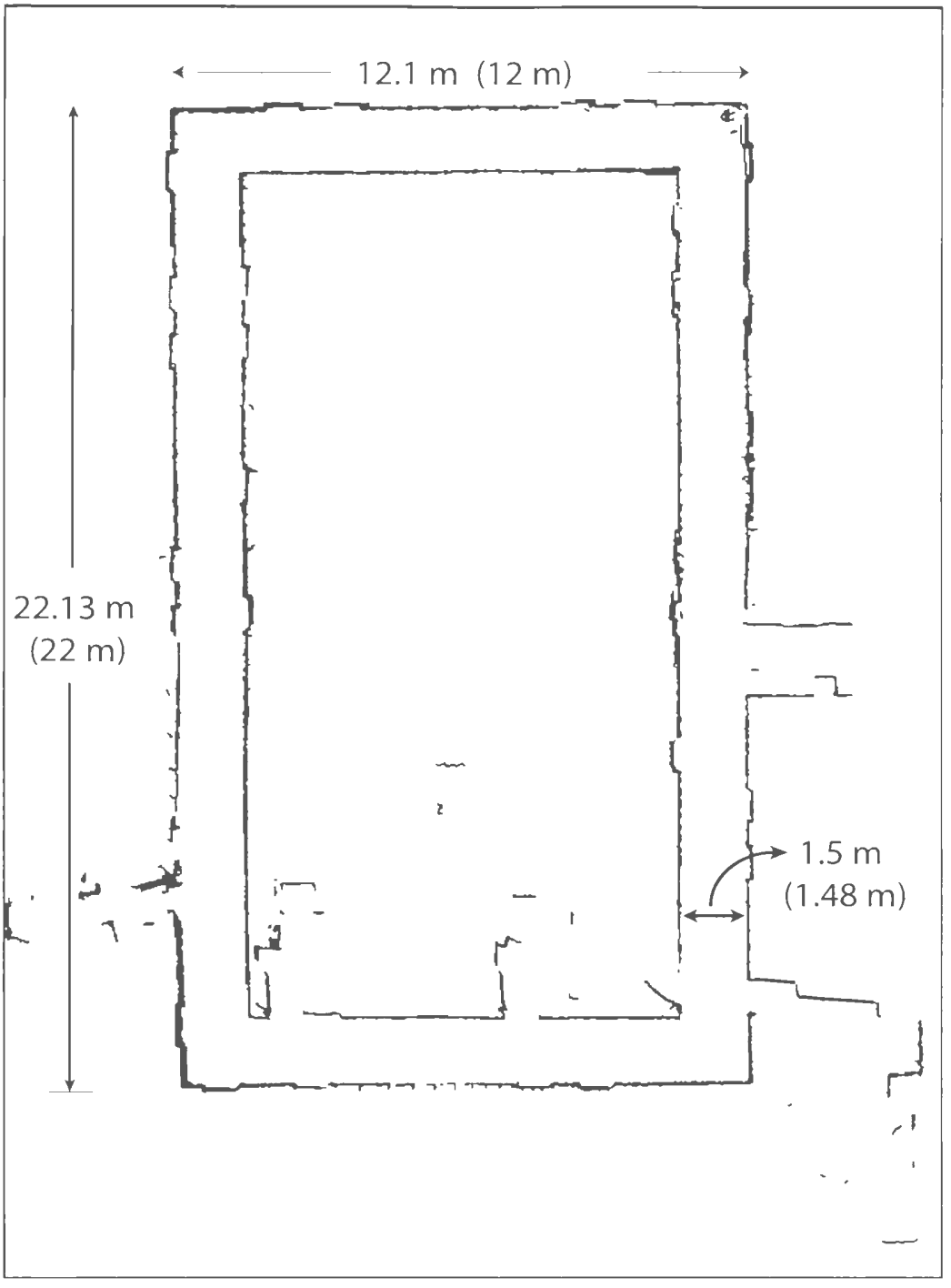


Figure 1.15: Mapping a large cyclic corridor of dimension $22m \times 12m$: Map generated using proposed CML algorithm. Quantities within bracket show actual dimensions. Other dimensions are according to the developed map

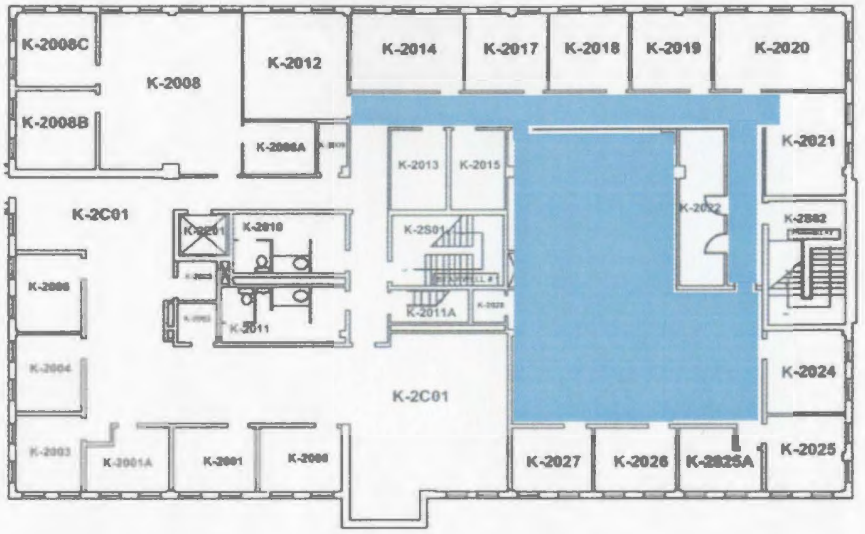


Figure 4.16: Blueprint of Level-2, C-CORE. Blue shaded area shows ISLAB ($17m \times 15m$)

red line shows odometry. The proposed algorithm successfully generates a topologically consistent map as shown in Figure 4.18.

4.3.5 Test 5:

- **Environment:** Approximately $34m$ long corridor. (blue print shown in Figure 4.20)
- **Surface:** Tile floor.
- **Range measurement:** After each $1m$ of travel or 30^o change in orientation.
- **Added odometry error:** High.

The corridor contains very few features for robot localization. The resulting map from odometry along with robot trajectory is shown in Figure 4.19. The final map (Figure 4.20) generated according to the proposed algorithm is topologically consistent and reflect the correct geometry of the corridor.

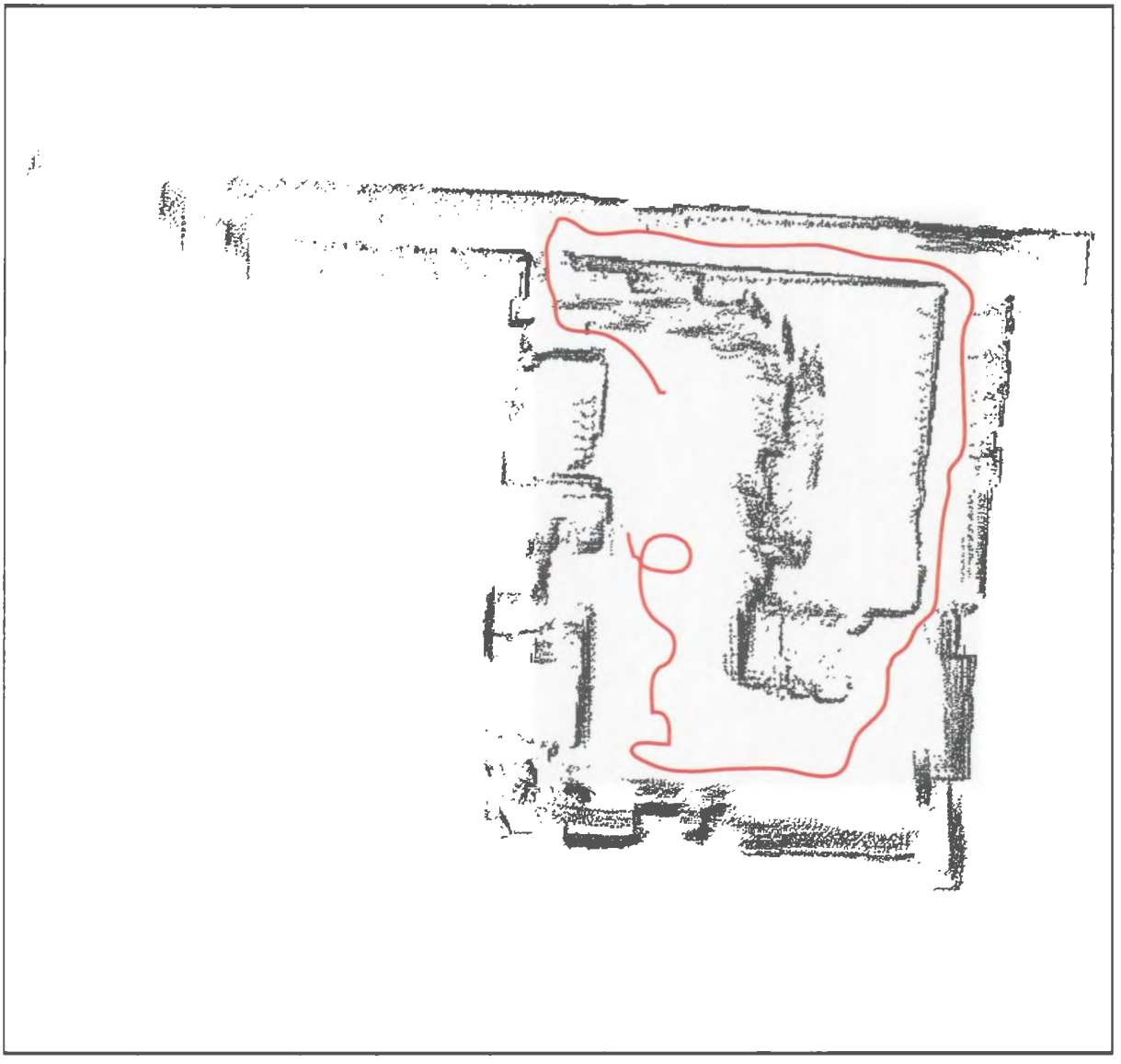


Figure 4.17: Mapping unstructured environment cyclic environment of dimension $17m \times 15m$: Map from odometry The red line shows robot trajectory according to odometry

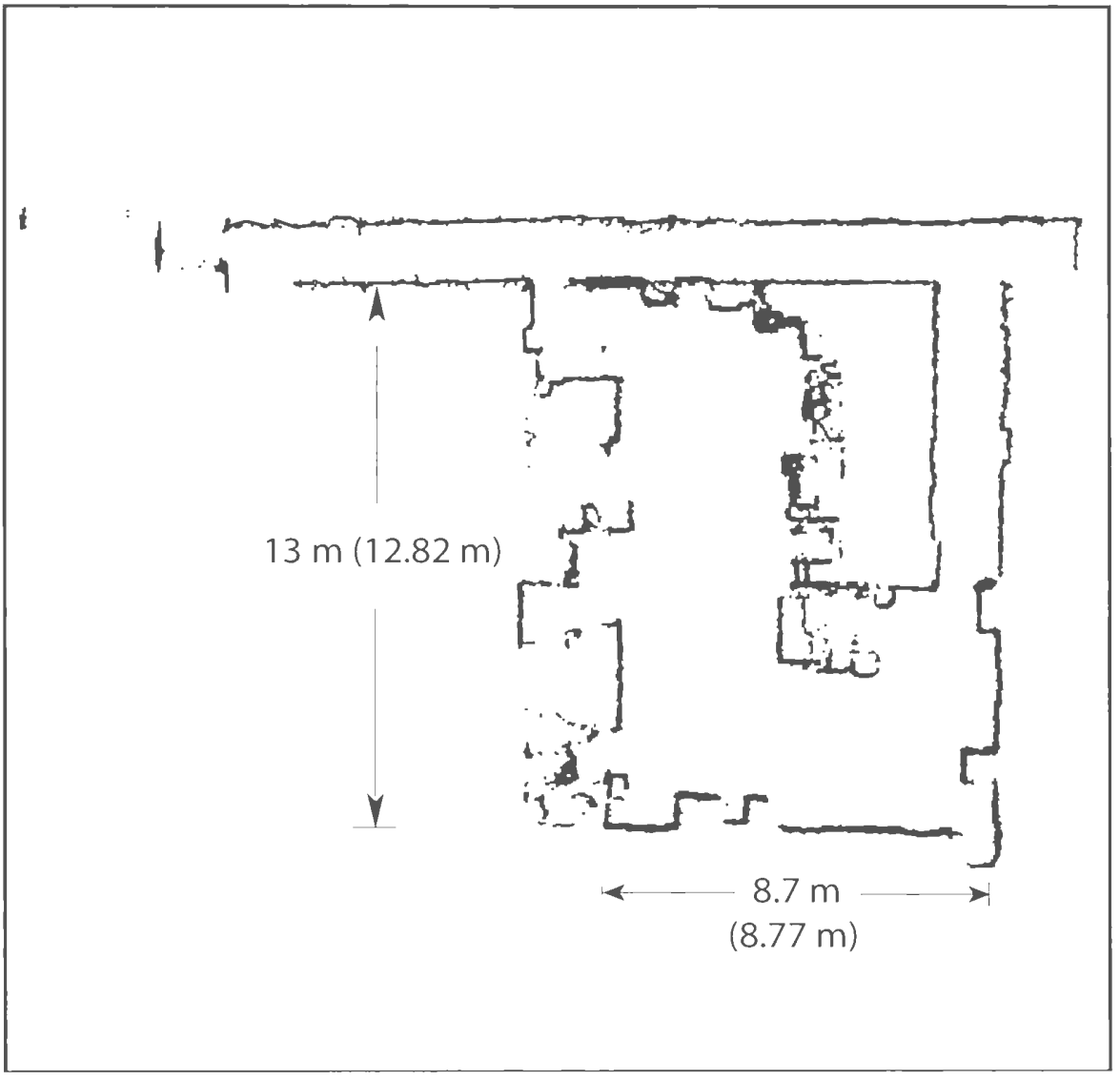


Figure 1.18: Mapping unstructured environment cyclic environment of dimension $17m \times 15m$: Map generated using proposed CML algorithm. Quantities within bracket show actual dimensions. Other dimensions are according to the developed map

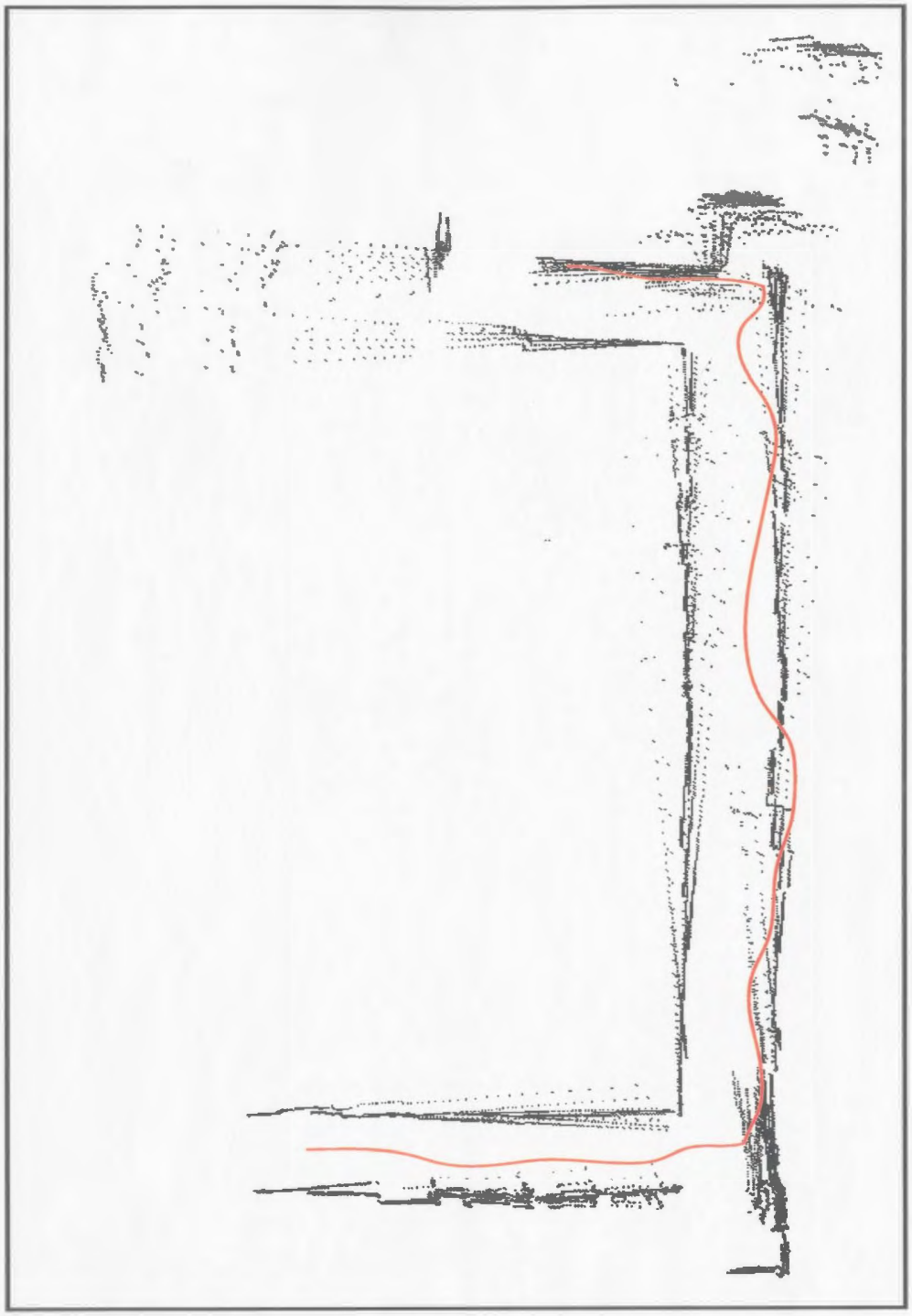


Figure 4.19: Mapping an approximately $34m$ long 'U' shaped corridor: Map from odometry. The red line shows robot trajectory according to odometry

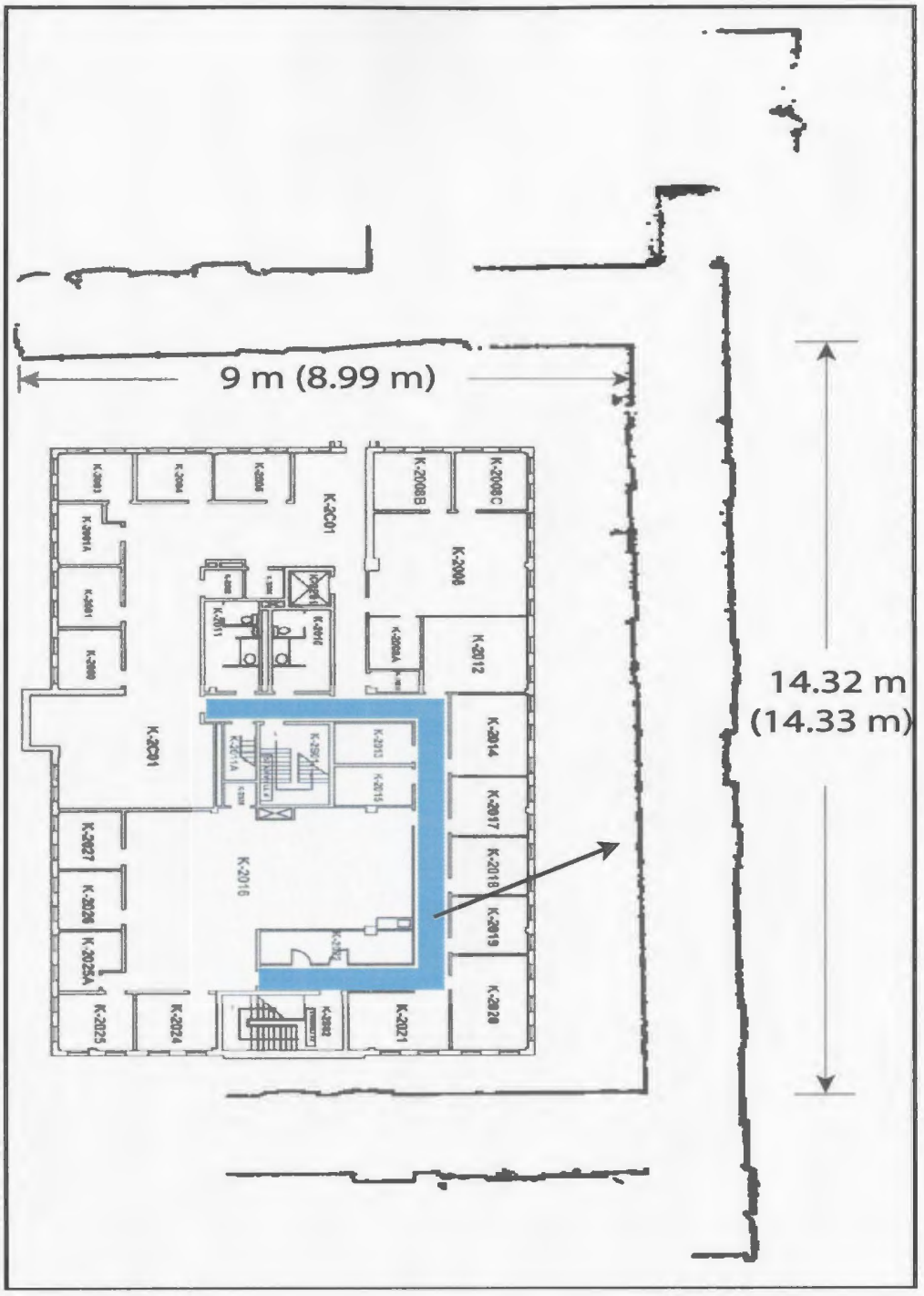


Figure 4.20: Mapping an approximately $34m$ long 'U' shaped corridor: Map generated using proposed CML algorithm. Blue shaded area in the blueprint shows the same corridor. Quantities within bracket show actual dimensions. Other dimensions are according to the developed map

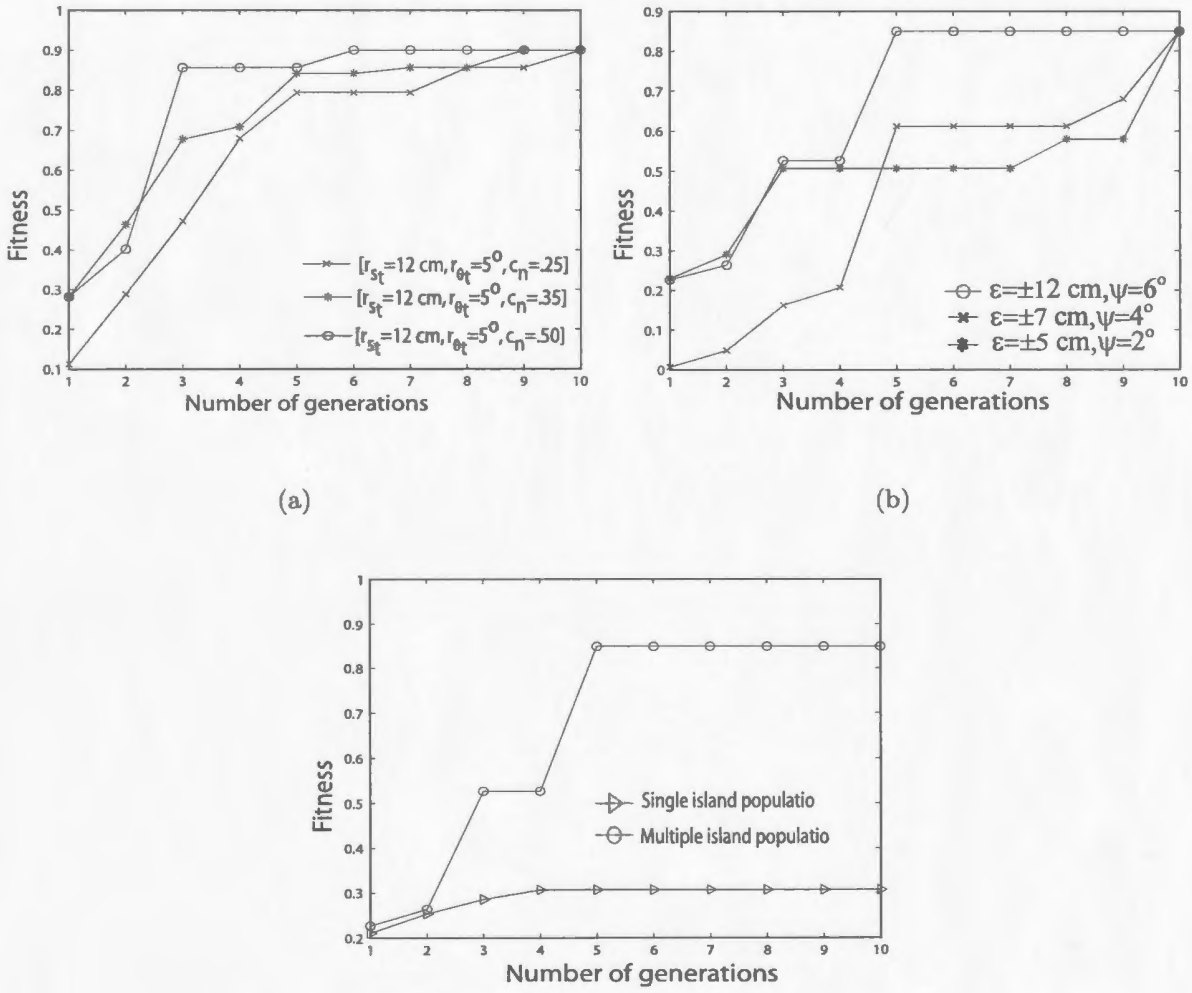


Figure 4.21: Convergence characteristics of the GA under parameters variation (a) Effect of sample density in $\hat{\mathbf{X}}_t$ (b) Effect of mutation parameters ψ and ϵ (c) Effect of multiple island model of population. Results are corresponding to the mapping example shown in Fig. 4.8

4.3.6 Convergence characteristics of the proposed algorithm

A set of parameters influences the convergence of the proposed Fuzzy-Evolutionary SLAM algorithm. This section provides a brief discussion about the effect of these parameters on the convergence characteristics of Fuzzy-Evolutionary SLAM.

Sample density in $\hat{\mathbf{X}}_t$ (determined by the value of c_n) has significant effect on convergence when $\mathbf{x}_t \in conv(\hat{\mathbf{X}}_t)$, though for $\mathbf{x}_t \notin \hat{\mathbf{X}}_t$ the effect becomes less obvious. For $\mathbf{x}_t \in conv(\hat{\mathbf{X}}_t)$ or $\mathbf{x}_t \in \hat{\mathbf{X}}_t$ high sample density (large c_n) ensures earlier convergence as the GA can detect/generate the correct solution within fewer number of generations. Fig. 4.21(a) shows the convergence characteristics of the GA with different values of c_n. Here the GA is involved in registering a local map of the second test environment for which $\mathbf{x}_t \in conv(\hat{\mathbf{X}}_t)$. Global maxima is at 0.95. For similar fuzzy prediction, $r_{s_t} = 12cm$ and $r_{\theta_t} = 5^o$, larger values of c_n cause the convergence to happen earlier. Results are corresponding to the island of population which contains the highest-fit chromosome in the termination generation.

The mutation parameters ϵ and υ play a vital role for convergence in global maxima when $\mathbf{x}_t \notin conv(\hat{\mathbf{X}}_t)$. This is because, an appropriate choice of ϵ and υ can generate a sample near to or within the basin of attraction of the global maxima which is outside of $conv(\hat{\mathbf{X}}_t)$. For the mapping example of Fig. 4.9 effect of different values of ϵ and υ on convergence is shown in Fig. 4.21(b). The output parameters of the *Fuzzy predictor* are $[r_{s_t} = 12cm, r_{\theta_t} = 5^o, c_n = 0.3]$. With larger values of ϵ and υ the GA can quickly generates samples near the correct basin of attraction and convergence occurs within fewer number of generations. But smaller values of ϵ and υ cause delayed convergence, even for too small values the algorithm may fail to reach the global maxima. The results shown in Fig. 4.21(b) are corresponding to island nine of the population.

Effect of multiple island model of population on the convergence of the algorithm is an interesting property to observe. Multiple island population model does not contribute much difference in result when $\mathbf{x}_t \in \hat{\mathbf{X}}_t$ or $\mathbf{x}_t \in conv(\hat{\mathbf{X}}_t)$. For $\mathbf{x}_t \notin conv(\hat{\mathbf{X}}_t)$ partial solutions in $\hat{\mathbf{X}}_t$ having relatively higher fitness win the competition and go to the next generations. These individuals, which are usually of same phenotype, generate offsprings essentially

similar to their parents. Thus the generation starts to loose diversity and ends up with premature convergence. Rather, with multiple island model as each island grows independently without interaction with other islands, the diversity in generation is maintained properly. Consequently, probability of reaching the correct basin of attraction of global maxima increases. For the mapping example of Fig. 4.9 the effect of multiple island population is shown in Fig. 4.21(c). In both cases the output parameters of the *Fuzzy Predictor* are $[r_{s_t} = 12cm, r_{\theta_t} = 5^o, c_u = 0.3]$. The GA with multiple island ($N_i = 6$) model converge to the global maxima while the GA with single population suffers premature convergence.

Chapter 5

Conclusion

5.1 Overview

This research exploits the intelligent properties of two soft computing techniques to develop a novel algorithm for robotic mapping. The algorithm uses both fuzzy logic and GAs to solve the CML problem of mobile robot. The proposed CML algorithm is an incremental mapping algorithm which addresses all the necessary requirements of robotic mapping. Odometry error modeling using fuzzy logic enables the algorithm to consider all the systematic and non systematic errors to infer the uncertainty in robot pose. A fuzzy error model of odometry generates a sample based prediction of the robot pose. A GA is designed to search for the optimal robot pose which can best accommodate a local map in a current partially developed global map. The GA based search starts from the pose space defined by the fuzzy error model of odometry. The property of *natural selection*, which encourages the better performing individuals to survive, offers an iterative solution to the correspondence problem of robotic mapping. The capacity of GA to both re-sample and re-parent new samples from the existing samples imposes less restriction on the initial sample distribution (provided by the fuzzy error model of odometry) to closely resemble the original distribution. This also enables the algorithm to avoid convergence to a local maxima.

5.2 Contributions to research

- **Combining soft computing methods for robotic mapping:** The proposed algorithm successfully combines soft computing methods, namely fuzzy logic and GA, to solve the CML of mobile robot. It utilizes the capacity of fuzzy logic to handle both qualitative and quantitative uncertainty while inferring the uncertainty in robot pose. Similarly, it uses the property of *natural selection* to iteratively solve the data association problem while mapping. Moreover, it uses the capacity of genetic operators (in re-parenting samples) to remove the restriction on the fuzzy error model to always include the true robot pose. Therefore, the proposed algorithm maintains a harmony between the two soft computing methods while applying them to robotic mapping. Combination of different soft computing techniques is a completely new concept in the literature of robotic mapping.

- **Fuzzy frame work for odometry error modeling:** The proposed algorithm uses fuzzy logic to model the errors in mobile robot's odometry. The existing odometry error modeling techniques use stochastic distribution to predict the uncertainty in robot pose introduced by several vehicle specific and real world parameters. The widely used stochastic model in this aspect is a two dimensional Gaussian distribution. This research proposes an alternative technique based on fuzzy rules to model the error in the robot's odometry. The proposed fuzzy inference model of odometry error provides a sample based prediction about the robot pose after execution of a control command. Effectiveness of fuzzy logic in odometry error modeling lies in the fact that a number of real world parameters influence the errors in odometry in a qualitative fashion, rather than in a quantitative way. Many of these parameters lack enough probabilistic sophistication to be modeled stochastically. Stochastic models of odometry, therefore, ignore these parameters and intend to model only the available quantitative uncertainty. Therefore, stochastic modeling of odometry error tends to be optimistic in certain environments where the qualitative parameters dominate in the odometry error (which is typical in outdoor environment). The

fuzzy logic has the ability to infer the available qualitative information into fuzzy quantitative form while avoiding the use of complex mathematical procedure. The proposed technique applies the knowledge of the robot's kinematics and its behavior at different environments to develop a fuzzy rule base and also to define various fuzzy subsets. The resulting fuzzy model of odometry error considers all the parameters related to odometry error(both qualitative and quantitative) in order to infer the uncertainty in robot pose. A key advantage of fuzzy based modeling of odometry error lies in its simplicity. As the rules of fuzzy logic are constructed from less restrictive axioms than probability theory, introducing new variables in the odometry model (due to change in robot's construction, environment conditions etc.) only requires the addition of one or more coherent mapping rules with the existing rule base. Fuzzy modeling of odometry error is a new concept in the literature of mobile robot's odometry and there remains much room for development.

- **GA based mapping:** A GA is used in the proposed algorithm to search for the best robot poses to produce a 'maximally consistent' map from the sensor measurements. After execution of each control command, the GA based search starts from the sample based prediction of the robot pose that has been predicted by the fuzzy error model of odometry. As the fuzzy prediction always 'remembers' the robot's pose errors at each step of its past moves, it usually includes the true robot pose. The genetic operators (mutation and crossover) are capable to re-parent samples outside the fuzzy sample based prediction. This imposes less restriction on the initial sample distribution to closely resemble the actual distribution of robot pose. Particle filter based CML algorithms usually require the initial sample distribution to include the correct basin of attraction. Therefore, under a poorly defined odometry error model, particle filter based CML converges to a local optima. The particle filters keep on updating the history of a set of samples upon receiving new sensor measurement. Therefore, they are only capable to re-sample from the initial sample distribution in order to support better performing samples, known as Sequential Importance Resampling (SIR) in literature [58]. The theory of GA, on the other hand, provides means of

both re-sampling and re-parenting from initial samples based on individual's fitness. This property has significant importance in the context of proposed algorithm, as the incorrect inference (from fuzzy error model) about the pose uncertainty does not lead the algorithm to premature convergence. The *natural selection* property that supports better performing samples to survive offers an iterative solution to data association problem of robotic mapping.

5.2.1 Further recommendations

Though the proposed robotic mapping algorithm is devised for both indoor and outdoor environment, its validity, in this research, has been tested only at different indoor environments. In addition, the present implementation of the algorithm follows the assumption that the robot is equipped with a terrain (or surface) recognition mechanism. Therefore, future works in the context of robotic mapping include:

- development of a real-time terrain recognition algorithm.
- construction of fuzzy subsets and fuzzy rules for different terrain conditions (such as rocky, sandy, grassy, etc).
- testing the performance of the algorithm at various outdoor locations.

Further investigation is also required to test the potential use of the algorithm for a navigating mobile robot employed in high level task planning and execution in an unknown environment.

Bibliography

[1] J. Bares and D. Wettergreen, "Dante ii: Technical description, results and lessons learned," *International Journal of Robotics Research*, vol. 18, no. 7, p. 621649, 1999.

[2] S. Thrun, D. Ferguson, D. Haehnel, M. Montemerlo, W. Burgard, and R. Triebel, "A system for volumetric robotic mapping of abandoned mines," in *Proc. of IEEE Int. Conf. Robot. Automat.*, 2003.

[3] D. Apostolopoulos, L. Pedersen, B. Shamah, K. Shillcutt, M.D.Wagner, and W. Whittaker, "Robotic antarctic meteorite search: Outcomes," in *IEEE Int. Conf. Robot. Automat.*, 2001, p. 11711179.

[4] C. Urmson, B. Shamah, J. Teza, M. Wagner, D. Apostolopoulos, and W. Whittaker, "A sensor arm for robotic antarctic meteorite search," in *Proc. of International Conference on Field and Service Robotics*, 2001.

[5] D. Bapna, E. Rollins, J. Murphy, M. Maimone, W. Whittaker, and D. Wettergreen, "The atacama desert trek: Outcomes," in *IEEE Int. Conf. Robot. Automat.*, 1998, p. 597604.

[6] H. Durrant-Whyte, S. Majumder, S. Thrun, M. de Battista, and S. Scheding, "A bayesian algorithm for simultaneous localization and map building," in *IEEE Int. Conf. Robot. Automat.*, 2003, pp. 131 138.

[7] M. P. Golombek, R. A. Cook, T. Economou, W. M. Folkner, A. F. Haldemann, P. H. Kallemeyn, J. M. Knudsen, R. M. Manning, H. J. Moore, T. J. Parker, R. Rieder, J. T. Schofield, P. H. Smith, and R. M. Vaughan, "Overview of the mars pathfinder

mission and assessment of landing site predictions," *Science*, vol. 278, no. 5344, p. 1743–1748, 1997.

[8] C. Thorpe and H. Durrant-Whyte, "Field robots," in *Proc. of the 10th International Symposium of Robotics Research (ISRR01)*, 2001.

[9] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE J. Robot. Automat.*, vol. 3, no. 3, pp. 249–265, 1987.

[10] ——, "Occupancy grids: A probabilistic framework for robot perception and navigation," Ph.D. dissertation, Carnegie Mellon University, 1989.

[11] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, vol. 9, no. 2.

[12] S. Thrun, "Robotic mapping: A survey," CMU, PA, Tech. Rep. CMU-CS-02-111, Feb. 2002.

[13] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots," *Machine Learning*, vol. 31, p. 2953, 1998.

[14] G. Dissanayake, H. Durrant-Whyte, and T. Bailey, "A computationally efficient solution to the simultaneous localization and map building problem," in *IEEE Int. Conf. Robot. Automat. Workshop W4: Mobile robot navigation and mapping*, 2000.

[15] J. Borenstein, B. Everett, and L. Feng, Eds., *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., 1996.

[16] W. Burgard, A. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artificial Intelligence*, vol. 114, no. 1-2, pp. 3–55, 1999.

[17] I. Cox and G. Wilfong, Eds., *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[18] D. Kortenkamp, R. Bonasso, and R. Murphy, Eds., *AI-based Mobile Robots: Case studies of successful robot systems*. MIT Press, 1998.

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

[20] R. Murphy, Ed., *Introduction to AI Robotics*. MIT Press, Cambridge, 2000.

[21] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE J. Robot. Automat.*, vol. 17, no. 3, pp. 229–241, 2001.

[22] Y. B-Shalom and X. Li, Eds., *Estimation and Tracking: Principles, Techniques, and Software*. MA: YBS, 1998.

[23] J. C. Cox, "A review of statistical data association techniques for motion correspondence," *Journal of Computer Vision*, vol. 10, no. 1.

[24] M. Montemerlo, "Fastslam: A factored solution to the simultaneous localization and mapping problem with unknown data association," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.

[25] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Proc. of the AAAI National Conference on Artificial Intelligence*, 2002.

[26] D. K. M. Montemerlo, S. Thrun and B. Wegbreit, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proc. of the International Joint Conference on Artificial Intelligence*, 2003.

[27] R. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," SRI, Tech. Rep. TR 1760 and 7239,, 1985.

[28] L.Moreno, J.M.Armingol, S.Garrido, A.Escalera, and M.A.Salichs, "A genetic algorithm for mobile robot localization using ultrasonic sensors," *Intelligent and Robotic Systems*, vol. 34, pp. 135–151, 2002.

[29] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE J. Robot. Automat.*, vol. 12, no. 6, pp. 869–880, 1996.

[30] S. Thrun, D. Fox, W. Burgard, and D. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, pp. 99–141, 2001.

[31] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *Journal of Robotics Research*,, vol. 20, no. 5.

[32] J. Borenstein and Y. Koren, "The vector field histogram–fast obstacle avoidance for mobile robots," *IEEE J. Robot. Automat.*, vol. 7, no. 3, p. 278288, 1991.

[33] J. Bulmann, W. Burgard, A. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, and S. Thrun, "The mobile robot rhino," *AI Magazine*, vol. 16, no. 1, 1995.

[34] D. Guzzoni, A. Cheyer, L. Julia, and K. Konolige, "Many robots make short walk," *AI Magazine*, vol. 18, no. 1, p. 5561, 1997.

[35] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "Probabilistic algorithms and the interactive museum tour-guide robot minerva," *International Journal of Robotics Research*, vol. 19, no. 11, p. 972999, 2000.

[36] B. Yamauchi and P. Langley, "Place recognition in dynamic environments," *Journal of Robotic Systems*,, vol. 14, no. 2, p. 107120, 1997.

[37] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous Robot Vehicles*, vol. 8, pp. 167–193, 1990.

[38] S. Thrun, "Exploration and model building in mobile robot domains," in *Proc. IEEE Int. Conf. Neural Networks*, 1993, pp. 175–180.

[39] S.Williams, G. Dissanayake, and D. Whyte, "Towards terrain-aided navigation for underwater robotics," *Advanced Robotics*, vol. 15, no. 5, 2001.

[10] F. Lu and E. Milios, "Globally consistent range scan alligment for environment mapping," *Autonomous Robots*, vol. 4, pp. 333–349, 1997.

[11] ——, "Robot pose estimation in unknown environments by matching 2d range scans," *Intelligent and Robotic Systems*, vol. 18, pp. 249–275, 1998.

[12] J.-S. Gutmann and C. Schlegel, "Amos: Comparison of scan matching approaches for self-localization in indoor environments," in *Proc. First European Workshop on Advanced Mobile Robots*, 1996.

[13] R. E. Kalman, "A review of statistical data association techniques for motion correspondence," *Journal of Basic Engineering*, vol. 82, p. 3545, 1960.

[14] P. Maybeck, Ed., *Stochastic Models, Estimation, and Control*. Academic Press, 1979.

[15] A. Jazwinsky, Ed., *Stochastic Processes and Filtering Theory*. NY: Academic, 1970.

[16] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robot," in *IEEE Int. Conf. Robot. Automat.*, 1999.

[17] S. Thrun, W. Burgard, and D. Fox., "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2000.

[18] J. Castellanos, J. Montiel, J. Neira, and J. Tardós, "The spmap: A probabilistic framework for simultaneous localization and map building," *IEEE J. Robot. Automat.*, vol. 15, no. 5, p. 948953, 1999.

[49] J. A. Castellanos and J. D. Tardós, Eds., *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*, ser. Lecture Notes in Statistics. Boston, MA: Kluwer Academic Publishers, 2000.

[50] P. Newman, "On the structure and solution of the simultaneous localisation and map building problem," Ph.D. dissertation, University of Sydney, 2000.

[51] A. P. Dempster, A. N. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *The Royal Statistical Society*, vol. 39, no. 1, pp. 1 38, 1977.

[52] W. Burgard, H. J. D. Fox, C. Matenar, and S. Thrun, "Sonar-based mapping of large-scale mobile robot environments using em," in *Proc. of the International Conference on Machine Learning*, 1999.

[53] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun, "Using em to learn 3d models of indoor environments with mobile robots," in *Proc. of the International Conference on Machine Learning*, 2001.

[54] B. Yamauchi, P. Langley, A. Schultz, J. Grefenstette, and W. Adams, "Magellan: An integrated adaptive architecture for mobile robots," Institute for the Study of Learning and Expertise, CA, Tech. Rep. 98-2, May 1998.

[55] B. Yamauchi and R. Beer, "Spatial learning for navigation in dynamic environments," 1996.

[56] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *IEEE Int. Conf. Robot. Automat.*

[57] B. Ristic, S. Arulampalam, and N. Gordon, Eds., *Beyond the Kalman Filter*. Boston, London: Artech House, 2004.

[58] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*. Boston, MA: Springer Verlag, 2001.

[59] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *Proc. IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2000.

[60] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastslam," in *IEEE Int. Conf. Robot. Automat.*

[61] S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, and E. Nebot, "Fastslam: An efficient solution to the simultaneous localization and mapping problem with unknown data association," *Journal of Machine Learning Research*, 2004.

[62] A. Eliazar and R. Parr, "Dp-slam: Fast, robust simulataneous localization and mapping without predetermined landmarks," in *Proc. of the International Joint Conference on Artificial Intelligence*, 2003.

[63] —, "Dp-slam 2.0," in *IEEE Int. Conf. Robot. Automat.*

[64] T. Ducket, "Concurrent map building and self localization for mobile robot navigation," Ph.D. dissertation, University of Manchester, 2000.

[65] C. Hein and A. Meystel, "A genetic technique for robotic trajectory planning," vol. 11, pp. 351–361, 1994.

[66] M. Chen and A. Zalzala, "Safety considerations in the optimization of paths for mobile robots using genetic algorithms," in *Proc. First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1995, pp. 299–304.

[67] J. Potvin, C. Duhamel, and F. Guertin, "A genetic algorithm for vehicle routing with backhauling," vol. 6, pp. 345–355, 1994.

[68] D. Kang, H. Hashimoto, and F. Harashima, "Path generation for mobile robot navigation using genetic algorithm," in *Proc. IEEE IECON*, vol. 1, 1995, pp. 167–172.

[69] J. M. Armingol, A. Escalera, L. Moreno, and M. Salichs, "Mobile robot localization using a non-linear evolutionary filter," *Advanced Robotics*, vol. 16, no. 7, pp. 629–652, 2002.

[70] J. M. Armingol, A. de la Escalera, L. Moreno, and M. Salichs, "A genetic algorithm for mobile robot localization using ultrasonic sensors," in *14th World Congress of IFAC International Federation of Automatic Control*, 1999.

[71] A. Neumaier, Ed., *Complete Search in Continuous Global Optimization and Constraint Satisfaction*, ser. Acta Numerica. Boston, MA: Cambridge University Press, 2004.

[72] E. Tunstel, H. Seraji, and A. Howard, *Intelligent Control Systems Using Soft Computing Methodologies*. LLC: CRC Press, 2001, ch. 11.

[73] H. Seraji, A. Howard, and E. Tunstel, "Terrain-based navigation of mobile robots: A fuzzy logic approach," in *Proc. Int. Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2001.

[74] I. Rekleitis, "Cooperative localization and multi-robot exploration," Ph.D. dissertation, McGill Universitiy, 2003.

[75] I. Rekleitis, G. Dudek, and E. Millios, "Probabilistic cooperative localization and mapping in practice," in *IEEE Int. Conf. Robot. Automat.*, 2003.

[76] J. Moon, C. Park, and F. Harashima, "Kinematic correction of differential drive mobile robot and a design for velocity trajectory with acceleration constraints on motor controllers," in *IEEE/RSJ Int. Conf. Intellegent Robot. Systems*, 1999, pp. 930–935.

[77] J. Yen and R. Langari, Eds., *Fuzzy logic: Intelligence, Control and Information*. Prentice Hall, 1999.

[78] L. A. Zadeh, "Discussion: Probability theory and fuzzy logic are complementary rather than competitive," *Technometrics*, vol. 37, no. 3.

[79] J. M. Mendel, Ed., *Uncertain rule-based fuzzy logic systems : introduction and new directions*. NJ: Prentice Hall, 2001.

[80] D. E. Goldberg, Ed., *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1989.

[81] Y. Leung, Y. Gao, and Z. Xu, "Degree of population diversitya perspective on premature convergence in genetic algorithms and its markov chain analysis," *IEEE Trans. Neural Networks*, vol. 8, no. 5, pp. 1165–1175, 1997.

[82] R. K. Ursem, "Diversity-guided evolutionary algorithms," in *Proceedings of Parallel Problem Solving from Nature*, pp. 462–471.

[83] A. Corcoran, "An overview of parallelism in genetic algorithms," The University of Tulsa, Tech. Rep., 1993.

[84] L. J. Eshelman, *The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination.* LLC: Morgan Kauffman, 1991, pp. 265–283.

[85] *Pioneer 3^{TM} Operations Manual*, ActivMedia Robotics, 2004.

[86] *Laser Measurement System*, 8 008 970/01-2002, SICK.