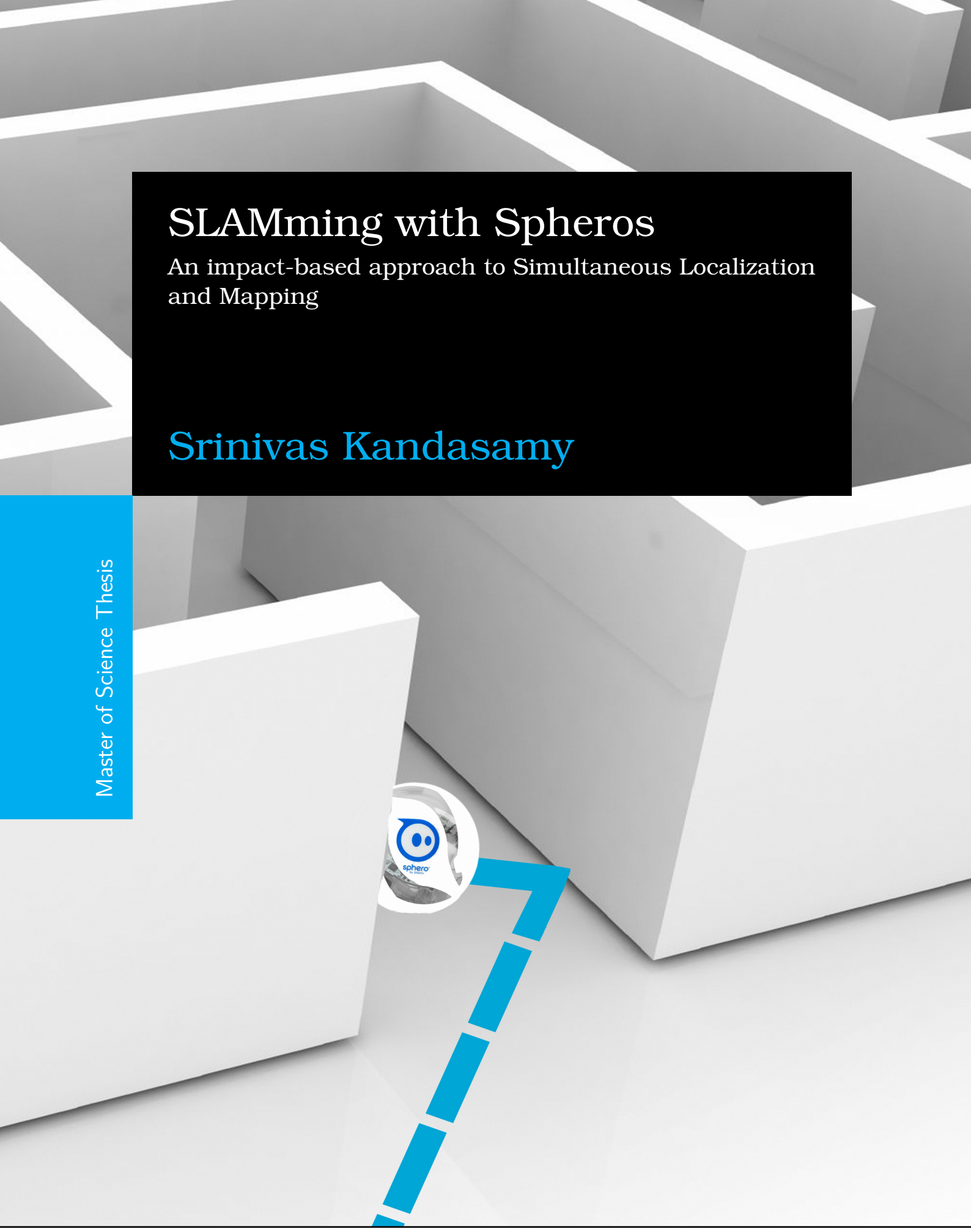
SLAMming with Spheros

An impact-based approach to Simultaneous Localization and Mapping

Srinivas Kandasamy

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

SLAMming with Spheros

An impact-based approach to Simultaneous Localization and Mapping

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Srinivas Kandasamy

August 13, 2015

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

The Simultaneous Localization and Mapping (SLAM) problem for mobile robots aims at consistently building a map of an unknown environment while simultaneously determining its position within this map. From a control-theoretic viewpoint, it is somehow analogous to simultaneously estimating the states and output map of the system. In the robotics community, SLAM is arguably considered a solved problem on a theoretical and conceptual level, but still it requires considerable maturity on a practical level. The state-of-the-art SLAM algorithms require computationally powerful processors, expensive sensors with dense feature extraction and multiple sensors for uncertainty reduction. An approach to the SLAM problem using minimal sensing information is still lacking in both theoretical and practical aspects.

In this context, this M.Sc. thesis aims to study and implement a special type of SLAM solely using the impact information from a spherical mobile robot called Sphero (developed by Orbotix). Such impact data is available from the robot's onboard Inertial Measurement Unit (IMU), however the impact angle and odometry information are subject to significant drift. Thus, the SLAM problem will be restricted to a rectilinear environment in order to allow for calibration and correction of such accumulated IMU errors (assuming that impacts with walls are sufficiently frequent).

The impact-based SLAM is an observable estimation problem with downside of a poor robot pose distribution. An accurate representation of the pose distribution is a particle set, with the resulting estimation technique as particle filter. Suitable map representations for the impact-based SLAM problem are formulated and studied, and the most efficient one is implemented. A probabilistic formulation is laid out for the SLAM problem using robot motion model and map representation, and associated challenges are studied for developing an efficient algorithm.

The resulting SLAM algorithm uses a Rao-Blackwellized particle filter which is computationally efficient and robust to data association errors. The issue of inconsistency is discussed for the developed SLAM algorithm and suitable modifications are proposed over the developed algorithm for ensuring consistency. SLAM is extended to multiple robots as a map-merging problem since multiple robots can build a perceptually rich map with a lower exploration time.

Table of Contents

List of Figures

Acknowledgements

I would like to thank my supervisor dr.ir. T. Keviczky for his guidance and continued enthusiasm which helped me complete this thesis. Your meticulous attitude to my work had helped me gain a deeper understanding on the subject, and meeting hours were especially invaluable. I also appreciate your swift response in getting the practical setup ready for my thesis work.

I would like to kindly acknowledge Frank De Winkel for his continued support on setting up the practical system. I would also like to thank Kees Slinkman for his insightful ideas and support in building a rectilinear environment.

Finally, I would like to thank my family and friends for their moral support during my stay in Netherlands.

Delft, University of Technology
August 13, 2015

Srinivas Kandasamy

Chapter 1

Introduction

The problem of Simultaneous Localization And Mapping, abbreviated as SLAM, is addressed as a fundamental problem in mobile robotics and is considered by many to be a prerequisite in truly autonomous robots. SLAM addresses the problem of a mobile robot moving through an unknown environment of which no map is available *a-priori*. The robot has to build a map of the environment through onboard sensing and its movements in the environment, both corrupted by noise. The goal of SLAM is to construct the map of an environment and path taken by the robot.

If the map is available *a-priori*, estimating the path of the robot would come close to a localization problem and similarly, if the true path of the robot is made available, building a map is relatively a simple task. However, if both the map and true path of the robot are unknown, correlations arise between the unknowns which constrains localization and mapping to be considered concurrently, and hence the name Simultaneous Localization And Mapping.

The thesis discusses about impact-based approach to SLAM problem and its associated solutions. The impact-based approach uses minimal sensing modalities to accomplish SLAM and the solution is implemented for a rectilinear environment using Sphero robots, developed by Orbotix Inc. The robot collides with walls of the environment and builds a map using the collision data and odometry information. The collision is detected through changes in accelerometer data and with this information, robot's orientation and odometry can define the pose of the landmark. This obstacle can represent a point landmark, a line since all the obstacle are straight walls. A map representation has to be defined for associating the measurement information with the map or in simple words, building the map of environment.

The suitability of a map representation depends primarily upon sensing modalities and operating environment. As a result, there do not exist a generic SLAM solution which can be suited to any environment and sensing modalities and at the same time produce a consistent solution. Hence, an algorithm has to be developed using the odometry and collision measurements to consistently estimate the map of a rectilinear environment and path taken by the robot. Moreover, achieving a SLAM solution using minimal sensing information has been less studied and this thesis provides a scope for it. These objectives had motivated me in the right direction to pursue this thesis.

This chapter provides a short survey of state-of-the-art in the field of SLAM. This chapter begins with the genesis of SLAM (Section 1-1), providing a short history of early developments in this field. Section 1-2 describes SLAM through an illustrative framework to gain a better understanding of the problem a detailed explanation of SLAM problem. With a good understanding of the problem, Section 1-3 puts forward the state-of-the-art solutions for the problem from an estimation viewpoint. These sections provide the appropriate background needed to understand the thesis. Section 1-4 advances the research objectives for this thesis work, followed by Section 1-5 laying out the key thesis contributions. Section 1-6 outlines a brief description of the work done in each Chapter of this thesis.

1-1 SLAM genesis

The genesis of the SLAM problem arose from two separate concepts, localization and mapping. The problem of localization has been long addressed since the Second World War under the name 'Tracking', while the mapping problem has a history from the geodetic mapping and recently, SLAM researchers are adopting techniques from the later [1]. The notion of a robot to concurrently map and localize grew out when probabilistic methods were just beginning to be introduced into both Robotics and Artificial Intelligence (AI). The SLAM problem was first brought up at the 1986 IEEE Conference on Robotics and Automation (ICRA) and a number of researchers recognized that consistent probabilistic mapping is very much required to accomplish SLAM.

The researchers initially focussed on assuming a series of approximations to the SLAM problem by reducing it to a decoupled localization-mapping problem. The researchers failed to address the convergence properties of the decoupled problem or the steady-state behaviour, and it was widely assumed that the errors in the map would not converge and would rather exhibit a random-walk behaviour.

The solution to SLAM addressed by [2] was the first conceptual breakthrough to provide a converging SLAM solution. The main aspect of this work addressed that the correlations between mapping and localization is important and the solution improved as correlations grew. The convergence and solution was detailed in a probabilistic framework and takes the uncertainty explicitly into account. The research from then on focussed mainly on challenges in implementation such as computational efficiency, data-association, nonlinearity, and consistency.

SLAM has now turned out to be an essential capability for mobile robots in unknown environments where globally accurate position data (e.g. GPS) is not available. In particular, significant promise has been shown for remote exploration, going to places that are too distant [3], too dangerous [4], or too costly for human access. If robots are to operate autonomously in extreme environments such as abandoned mines or extra-terrestrial navigation, they must be capable of building accurate maps and navigating reliably according to these maps. Even in benign environments such as indoor environments, accurate prior maps are difficult to acquire. The applications of SLAM, listed so far, makes a primary assumption that the environment is static in nature. However, the extension of SLAM to dynamic environments is not straightforward and is more complex. Recent literature [5] has shown SLAM to work in dynamic environments under various assumptions for human-robot interaction.

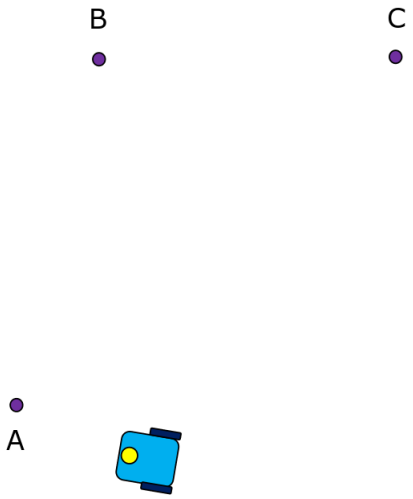


Figure 1-1: A simple environment for SLAM[1]

1-2 SLAM problem

Before stepping into mathematical formulations of SLAM, Example 1.1 gives an illustration of a general SLAM problem.

Example 1.1. *Consider a mobile robot, equipped with a range sensor and an odometry sensor, placed in an unknown environment. The robot has to implement SLAM by exploring the environment using the given sensors.*

A simple environment is taken into consideration which contains three distinct point landmarks as shown in Figure 1-1. The range sensor provides information about the landmark location in the environment while the odometry sensor is used to track the position of the robot through its movements. The measurements obtained from both the sensors are corrupted with noise, and for simplicity, the uncertainty in the measurement is modelled as a Gaussian. Here the robot is assumed to detect the corresponding landmark from the measurement, but in future, this problem (called as data association) will considered in detail. The robot at its initial location (trivially taken as origin of the map to built), as shown in Figure 1-2, observes landmark A in its field of view and takes a range measurement. The uncertainty associated with this measurement is represented by an ellipse as shown in Figure 1-3, which is a 2D Gaussian.

The robot begins to explore the environment after the update of landmark A into its memory. The uncertainty associated with its motion measurement from odometry is represented by a similar Gaussian as shown in Figure 1-4. Due to the uncertainty in the robot's position or the path of the robot, subsequent range measurements of landmarks B and C are corrupted gradually as shown in Figure 1-5 and Figure 1-6. Hence, the map built through the robot's sensors begin to drift due to this uncertainty.

[1]Courtesy of Margarita Chli for the SLAM example figures

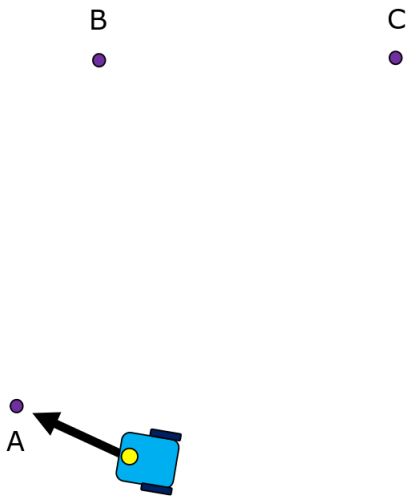


Figure 1-2: First measurement of landmark A

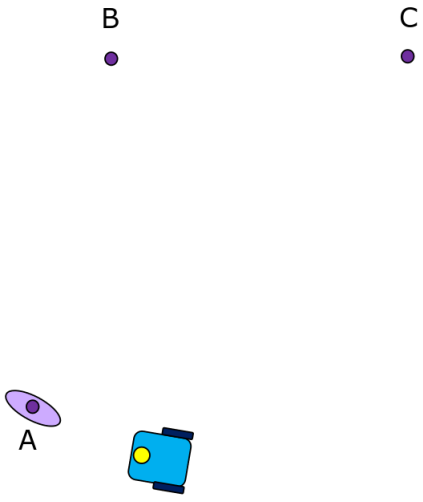


Figure 1-3: Update of landmark position A

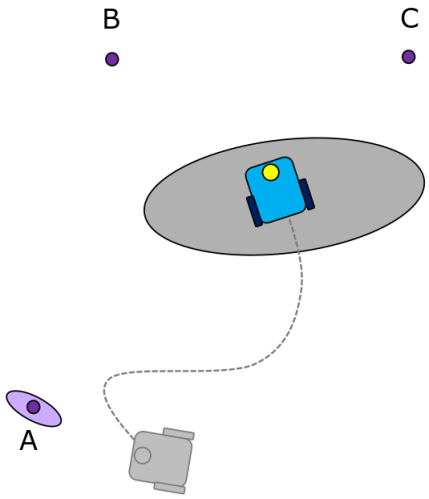


Figure 1-4: Exploration of the environment

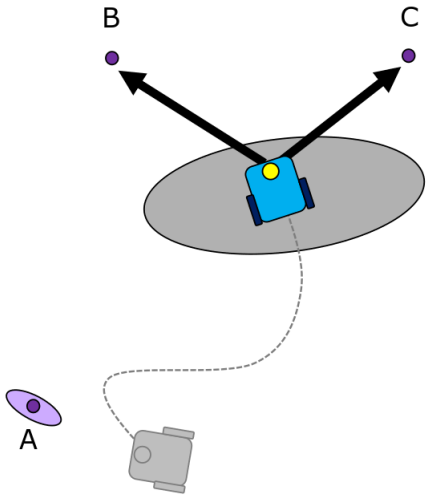


Figure 1-5: Measurements of landmarks B and C

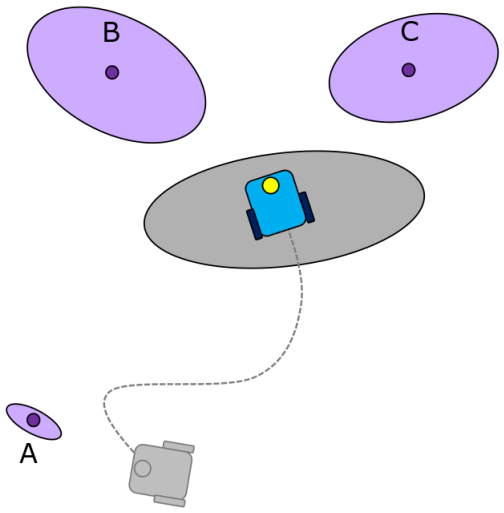


Figure 1-6: Update of landmarks B and C

The robot continues to explore with the error in the robot position or path begins to grow systematically. At certain point in time, the robot detects an old landmark as shown in Figure 1-7. The robot compares a new measurement of that landmark with the old measurement and finds a mismatch between them. The main factor contributing to this problem is the error in the robot's current position which is corrected using the new measurement and the old updated estimate of landmark A. Since the entire set of landmark measurements were correlated due to the error in robot's path, all the landmarks get updated as a result. This step is especially called as loop closure problem and it is illustrated in Figure 1-8. □

The illustration of SLAM makes the problem more clear and gives an intuitive idea. A key factor affecting the estimates of the landmark positions was the error in the robot position or path and this factor bridges a correlation between the landmark estimates and robot position. Moreover, unlike the measurement noise, the error in the robot's pose has a systematic effect on the error in the map and as a result, the true map cannot be estimated without estimating the true path of the robot. This property gives rise to the chicken-egg nature of the SLAM problem.

A probabilistic formulation of the SLAM problem is used for developing the SLAM algorithm which explicitly deals with the uncertainty in the problem. The probabilistic approach has received wide-spread attention due to their applicability in SLAM and has contributed to a separate dimension in the field of robotics called Probabilistic Robotics [6].

According to standard SLAM formulation, a robot executes control and accumulates observations, both corrupted by noise. Each control or observation, coupled with an appropriate noise model, can be thought of as a probabilistic constraint. As the robot explores through the environment, the network of constraints expand and get updated with re-observations. With re-observations, the constraints turn less uncertain and become increasingly rigid. In the limit of infinite observations and controls, the position of all map features become fully correlated. The SLAM problem is illustrated as a network of constraints in Figure 1-9. For the

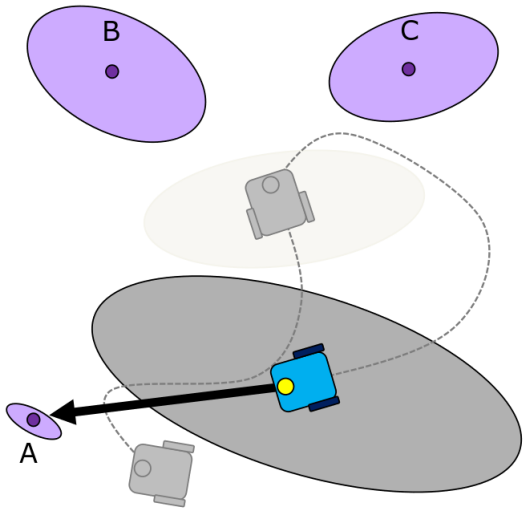


Figure 1-7: Second measurements of landmark A

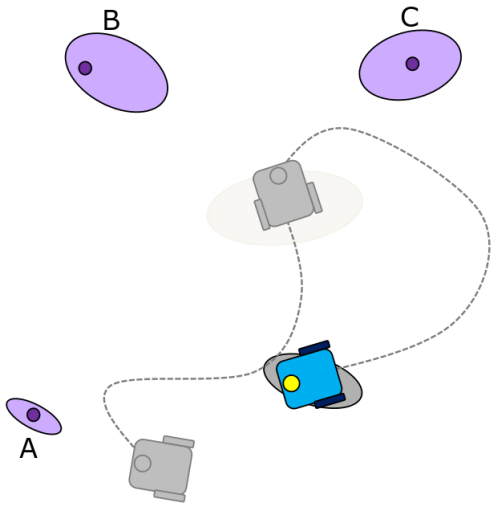


Figure 1-8: Loop closure in SLAM

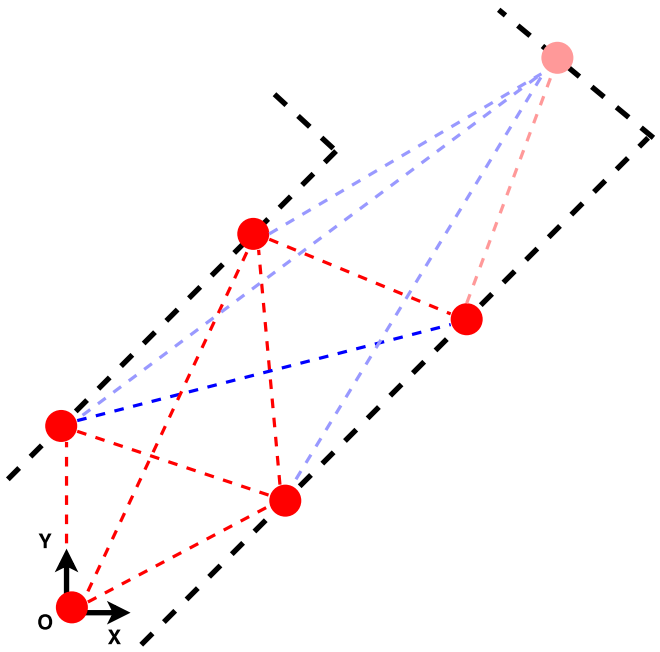


Figure 1-9: Network of constraints in impact-based SLAM; multiple observations of landmarks (dark red color) are constrained higher than the new landmarks (light red color). The blue lines represent the correlations through environment structure.

case of structured environments like the rectilinear environment in the Figure, the landmarks are additionally correlated through geometric structure such as orthogonality.

A popular extension of the above formulation is posterior estimation of the SLAM problem. The objective is to estimate a probability distribution over the possible maps and robot poses, given the observations and controls. The distribution, referred to as SLAM posterior, can be written as:

$$p(s_t, m | z_{1:t}, u_{1:t}, c_{1:t}) \tag{1-1}$$

where s_t is the robot pose at time t and m is the map of an environment. The variables $z_{1:t}$, $u_{1:t}$ and $c_{1:t}$ corresponding to the set of measurements, controls and data associations respectively and the posterior is conditioned upon them. The data association describes the mapping between a measurement z_t and a landmark in m. The description of these variables will be explained in detail in the forthcoming chapter.

There does exists an alternative formulation which is suitable for computing the best possible map and robot pose. This approach is commonly called an offline optimization approach where a global nonlinear optimization is carried over a batch of measurements. This approach has gained popularity in the recent years by exploiting the sparsity in SLAM.

The posterior approach seems desirable since it gives a distribution of possible solutions along with the uncertainty associated with it. This approach is robust in noisy environments since it explicitly considers the uncertainty factor in the problem. At first glance, the optimization approach might seem simpler than the posterior approach, however by considering judicious assumptions about how the state of the world and robot evolves, the posterior computation

can be more efficient. For example, consider the following recursive formula, known as Bayes filter, which computes the posterior at time t.

$$\begin{aligned} p(s_t, m|z_{1:t}, u_{1:t}, c_{1:t}) \;&=\; \eta\, p(z_t|s_t, m, c_t)\cdot \\ &\cdot \int p(s_t|s_{t-1}, u_t)\, p(s_{t-1}, m|z_{1:t-1}, u_{1:t-1}, c_{1:t-1}) ds_{t-1} \end{aligned} \tag{1-2}$$

The above integrand cannot be computed in closed form for a general probability distribution. However, if a particular form of distribution is considered such as Gaussian, a closed loop solution is indeed possible. Variants of Bayes filter such as Kalman filter and Particle filter are discussed in forthcoming section.

1-3 State-of-the-Art solutions

Achieving a solution to the SLAM problem was one of the most demanding research in mobile robotics literature for past two decades. SLAM has been regarded as a prerequisite for a robot to be truly autonomous and from decades of research, a practical solution is found to be indeed possible.

A consistent solution is achieved only when a complete set of correlations are propagated through a sequence of measurements and controls. However, the efficiency of computation ($\mathcal{O}(n^2)$, n being the state dimension) became an obstacle for achieving a practical solution. Researchers [5] later came up with more robust SLAM algorithms which are practically efficient in terms of computational and memory requirements through exploitation of structure in SLAM. This estimation technique is called as online estimation since measurements are processed as they arrive.

The two state-of-the-art approaches for SLAM using online estimation techniques are Extended Kalman Filter-SLAM and Particle filter-SLAM. The two approaches are formulated and discussed in the following sections.

1-3-1 Extended Kalman filter-SLAM

Extended Kalman Filter (EKF) is a parametrized representation of Bayes filter which approximates SLAM posterior as a multivariate Gaussian distribution over all features in the map and robot pose. The parametrized representation describes the probability distribution uniquely using mean μ and covariance Σ.

The state transition probability is described using the motion model of the robot,

$$p(s_t|s_{t-1}, u_t) \Longleftrightarrow s_t \;=\; f(s_{t-1}, u_t) \;+\; w_t, \tag{1-3}$$

where $f(\cdot)$ corresponds to the robot's motion model, s_t and u_t represent the robot pose and control input respectively at time instant t, and w_t represents an additive, zero-mean, uncorrelated Gaussian noise with covariance Q_t.

In a similar manner, the observation model is described using the following form-

$$p(z_t|s_t, m) \Longleftrightarrow z_t \;=\; h(s_t, m) \;+\; v_t, \tag{1-4}$$

where $h(\cdot)$ describes the geometry of the observation, z_t denotes observation and v_t represents another additive, zero-mean, uncorrelated Gaussian noise with covariance R_t.

The state vector for EKF-SLAM is usually defined as an augmented state of robot pose s_t and observed landmarks m. For the observation of a new landmark, a new state is defined using the inverse of observation model, and then append to the original state vector. A feature-based representation is used here where the individual observed point landmarks are defined either through their location or pose. Hence, a map is represented as a set of landmark positions or poses, $m = \Big[m_1, m_2, \cdots, m_N\Big]$. Additional details on map representation will be discussed in Section 2-4.

The standard EKF method can be applied to compute the mean μ_t and the covariance P_t given the measurements $z_{0:t}$ using a recursive update as shown below.

Prediction step

$$\hat{s}_{t|t-1} = f(\hat{x}_{t-1|t-1}, u_t) \tag{1-5}$$

$$P_{t|t-1} = \nabla f \cdot P_{t-1|t-1} \cdot \nabla f^T \; + \; Q_t \tag{1-6}$$

The gradient ∇f is the Jacobian of f evaluated at the estimate $\hat{x}_{t-1|t-1}$.

Update step

$$\begin{bmatrix} \hat{x}_{t|t} \\ \hat{m}_t \end{bmatrix} = \begin{bmatrix} \hat{x}_{t|t-1} \\ \hat{m}_{t-1} \end{bmatrix} + K_t \cdot \Big(z_t - h(\hat{x}_{t|t-1}, \hat{m}_{t-1})\Big) \tag{1-7}$$

$$P_{t|t} = P_{t|t-1} - K_t \cdot S_t \cdot K_t^T \tag{1-8}$$

$$S_t = \nabla h \cdot P_{t|t-1} \cdot \nabla h^T + R_t \tag{1-9}$$

$$K_t = P_{t|t-1} \cdot \nabla h^T \cdot S_t^{-1} \tag{1-10}$$

The Kalman gain is denoted by K_t and innovation covariance is denoted by S_t. The Jacobian $\nabla(\cdot)$ is evaluated with respect to state estimates at respective time instance.

The above SLAM algorithm assumes that the correspondences of all the landmarks are known prior. SLAM algorithm with unknown correspondences will be detailed in Section 2-5-3, which is a small addition to the existing EKF-SLAM algorithm. Moreover, it is assumed here that all the landmarks are initialized prior.

The prediction step propagates the mean and covariance of the belief in accordance to the motion model. This propagation step only affects the belief distribution associated with the robot pose. The landmarks of the map retain the same mean and covariance due to the static world assumption.

In the update step, the observed landmark is updated using the innovation and Kalman gain. The Kalman gain is a matrix of size 3 by $3N+3$, N being the number of landmarks. This matrix is usually not sparse and the information is propagated through the entire state estimate. The fact that the Kalman gain is fully populated for all state variables and not just for the observed landmark and robot pose is because of the correlations, that is observing the landmark does not just improve the position estimate of this very landmark but all the other landmarks as well. The situation is neatly depicted through the Example 1.1. Issues such as convergence, computational efficiency and data association have been addressed extensively in literature and a good survey discussing these issues is [7]. The issues specific to impact-based SLAM problem is discussed in Section 2-5.

1-3-2 Particle filter-SLAM

With the above parametric form of estimation, a closed-form solution is obtained for SLAM. However, various issues such as computational efficiency and data association raised questions of implementing EKF-SLAM for large-scale environments. Meanwhile, the success of non-parametric estimation techniques such as particle filters and its variant, Markov Chain Monte Carlo (MCMC) have emerged as a core algorithm in Machine Learning. The direct applicability of particle filter to SLAM is intractable (Remark 1.1) since they are subjected to the curse of dimensionality and a straightforward implementation of particle filter algorithm will be intractable due to the huge number of variables involved in the map.

Remark 1.1. *The computational requirement scales exponentially with the number of dimensions in a particle filter, as opposed to quadratic scaling of computations for a Gaussian update in an EKF.*

The conceptual breakthrough in the success of particle filters in SLAM was developed by [8]. The probabilistic Markov chain property and conditional independence were best exploited in implementing a tractable version of particle filter-SLAM. Specifically, the full SLAM problem with known correspondences possess a conditional independence property between any two disjoint set of features in the map, given the robot trajectory. In other words, if the robot trajectory is known to be accurate, the features or landmarks can be estimated independently. The correlation property stressed through the covariance matrix in Gaussian filters is exploited here as a conditional independence relation. This structural observation will make it possible to apply a specific version of particle filters known as Rao-Blackwellized Particle Filter (RBPF) to the SLAM problem. The RBPF version will be hereby simply addressed as particle filter-SLAM. The effect of this exploitation brings the tractability to the problem. Note that bringing tractability through approximation has their own downsides. For example, particle resampling will have a long term effect on the consistency of the filter [9] since the process noise (robot pose error) correlates the landmark estimates unlike standard estimation techniques where the error fades out with incorporation of measurements.

A naive implementation of the RBPF-SLAM requires $\mathcal{O}(MN)$ time, where M is the number of particles in the filter and N is the number of landmarks. A tree-based data structure of landmark representation can reduce the time complexity and memory requirements from linear scale to logarithmic scale $\mathcal{O}(M \log N)$, making it significantly faster than the state-of-the-art EKF-SLAM algorithm [10]. This had led to the success of particle filter-SLAM for a real-time implementation in large environments, and its contribution to Stanley car, DARPA [11]. Another useful property of particle filter-SLAM is data association decisions which is made on per-particle basis rather than the most likely particle. As a result, the particle filter can maintain posterior over multiple associations and helps approximating the full posterior $p(s_{1:t}, m, c_{1:t}|z_{1:t}, u_{1:t})$. This property of RBPF-SLAM makes it robust towards data association errors.

Apart from the above favourable properties, ability of particle filter to represent an arbitrary distribution is a generic property and can be usefully exploited here. A non-Gaussian distribution can be easily represented through samples, and hence nonlinear motion models can be easily implemented even in situations when the pose uncertainty is known to be very high.

Assuming a SLAM posterior distribution with known correspondences, the SLAM problem

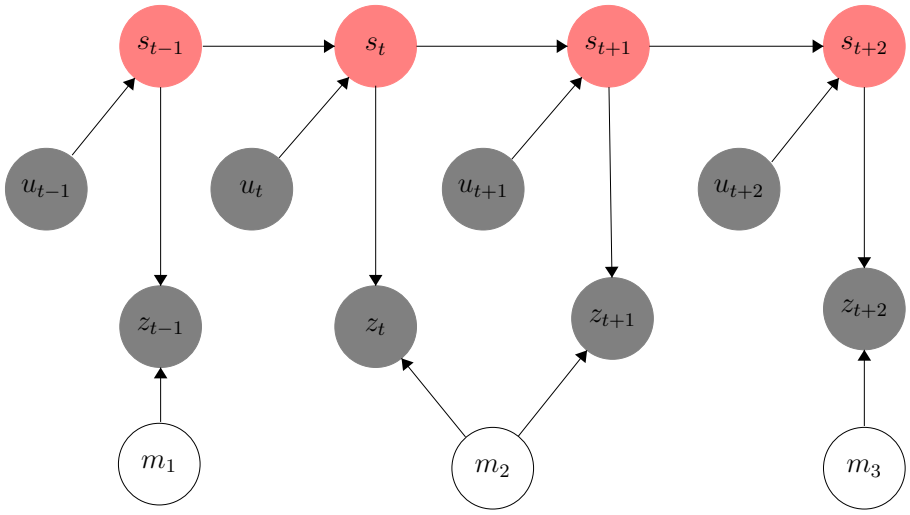


Figure 1-10: SLAM problem as a Bayes network graph. The control and measurements (gray shaded nodes) render conditional independence in mapping once the entire robot path (red shaded nodes) is known.

can be formulated as a joint posterior,

$$p(s_{1:t}, m|z_{1:t}, u_{1:t}) \tag{1-11}$$

The assumption of known correspondences takes into account of perfect data association decisions taken *a-priori*. In case of unknown correspondences, the joint posterior includes another variable $c_{1:t}$ which is estimated using data association techniques.

The full SLAM posterior can be factorized based on the conditional independence property or Rao-Blackwellization.

$$p(s_{1:t}, m|z_{1:t}, u_{1:t}, c_{1:t}) = p(s_{1:t}|z_{1:t}, u_{1:t}, c_{1:t}) \cdot \prod_{n=1}^{N} p(m_n|s_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \tag{1-12}$$

This factorization leads to a problem of estimating posterior over the robot path and N problems of estimating N landmark locations conditioned on the robot trajectory. The factorization is exact and it is based on the assumption that the robot trajectory is known to be accurate. The independence assumption also holds valid when the landmarks are randomly placed in the environment or the environment is unstructured. The factorization in the particle filter-SLAM is illustrated in Figure 1-10.

The path estimate $p(s_{1:t}|z_{1:t}, u_{1:t}, c_{1:t})$ can be computed efficiently from the sample space since a good approximation of the posterior can be found even in the presence of nonlinear motion models. The landmarks are separately estimated using low dimensional EKFs, and since the landmark estimates are conditioned on the robot path, each particle in the particle filter has its own map of landmark estimates.

The particle filter-SLAM algorithm maintains a set of particles, Y_t, of size M and the k^{th} particle is denoted as $Y_t^{[k]}$. The k^{th} particle is described systematically as a robot pose (or path) estimate $s_t^{[k]}$ augmented with a set of Gaussian feature estimates as shown below.

$$Y_t^{[k]} = \left\langle s_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle \tag{1-13}$$

Path estimation

The particle filter-SLAM samples a new pose s_t in accordance with the k^{th} particle from the motion model,

$$s_t^{[k]} \sim p(s_t | s_{t-1}^{[k]}, u_t) \tag{1-14}$$

where u_t is the current control action and $s_{t-1}^{[k]}$ is the posterior estimate of the robot pose at time $t-1$, residing in the k^{th} particle. The resulting sample is appended to a temporary particle set which contains the propagated particles at time t.

As the previous particle set Y_{t-1} is distributed according to the posterior at $t-1$, the temporary particle set is distributed according to $p(s_t | z_{t-1}, u_t, c_{t-1})$. This distribution is referred to as proposal distribution in particle filter.

After generating M new particles this way, the new posterior set Y_t is obtained from the temporary set by drawing particles (with replacement) with a probability proportional to its importance weight,

$$w_t^{[k]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(s_t^{[k]} | z_{1:t}, u_{1:t}, c_{1:t})}{p(s_t^{[k]} | z_{t-1}, u_t, c_{t-1})}. \tag{1-15}$$

The importance weights for all the particles account for the mismatch between the proposal and the target distribution. This can be illustrated in Figure 1-11. The exact calculation of this importance weight will be discussed below.

Note that only the latest pose estimate is used to compute the future posterior in Equation 1-15 according to Markov assumption (see Definition 2.1). The improvement in computational efficiency as a result of this assumption will be seen in the forthcoming chapter.

Landmark location estimation

From the particle description (1-13), each particle maintains a set of Gaussians (mean and covariance estimate) for landmarks in the map. For planar environment with point landmarks, the mean $\mu_{i,t}^{[k]}$ is a 2-element vector and the covariance $\Sigma_{i,t}^{[k]}$ is a 2×2 matrix for a landmark. These variables are propagated with the incoming measurements.

For a landmark whose correspondence does not relate to the measurement, the corresponding Gaussian is unchanged.

$$\left\langle \mu_{i,t}^{[k]}, \Sigma_{i,t}^{[k]} \right\rangle = \left\langle \mu_{i,t-1}^{[k]}, \Sigma_{i,t-1}^{[k]} \right\rangle \tag{1-16}$$

Otherwise, the corresponding Gaussian is updated using Bayes' product.

$$p(m_{c_t} | s_{1:t}, z_{1:t}, c_{1:t}) = \eta \cdot p(z_t | s_t, m_t, c_t) \cdot p(m_{c_t} | s_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \tag{1-17}$$

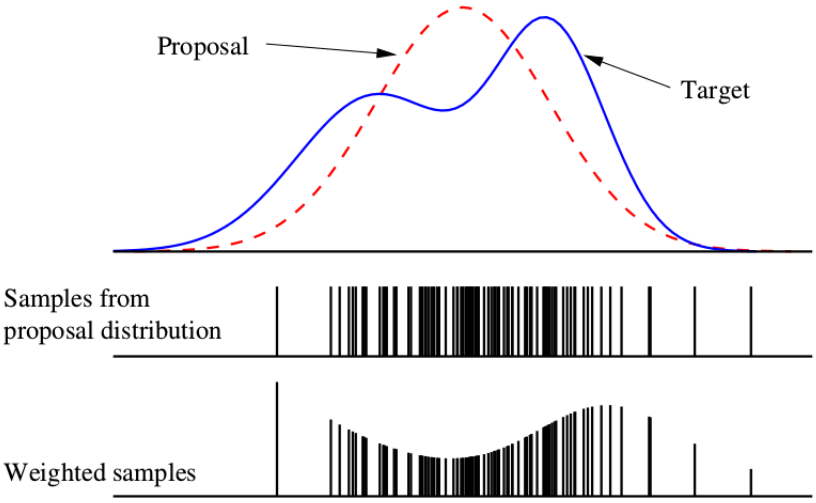


Figure 1-11: Importance Sampling step in a particle filter [6]

This update is implemented using an EKF where the observation model $z_t = h(s_t, m)$ is linearized at the current system state.

Calculation of importance weights

The problem of calculating the importance weights is continued from (1-15) as follows,

$$w_t^{[k]} \propto \frac{p(s_t^{[k]}|z_{1:t}, u_{1:t}, c_{1:t})}{p(s_t^{[k]}|z_{t-1}, u_t, c_{t-1})} \tag{1-18}$$

From Bayes' product,

$$\propto p(z_t, c_t|s_t^{[k]}, z_{1:t-1}, u_{1:t}, c_{1:t-1}), \tag{1-19}$$

and using Markov assumption,

$$\approx \int p(z_t|m_i^{[k]}, s_t^{[k]}, c_{1:t}) \cdot p(m_i^{[k]}) \cdot dm \tag{1-20}$$

The above integral can be computed as a finite sum due to a discrete representation of a map. The probabilities can be convolved as Gaussians using a linearized observation model and a Gaussian posterior of the map. For a detailed proof of the importance weight calculation, refer [8].

In the above formulation, the particles are sampled according to the motion model using the current control input and then weighted according to its likelihood with the incoming measurement. The particles are then resampled to obtain the posterior distribution. One of the practical issues in SLAM is degeneracy in evolution of posterior distribution over time. The degeneracy is observed in a scenario where the motion model is less accurate as compared to the observation (measurement) model. The proposal distribution using the less accurate

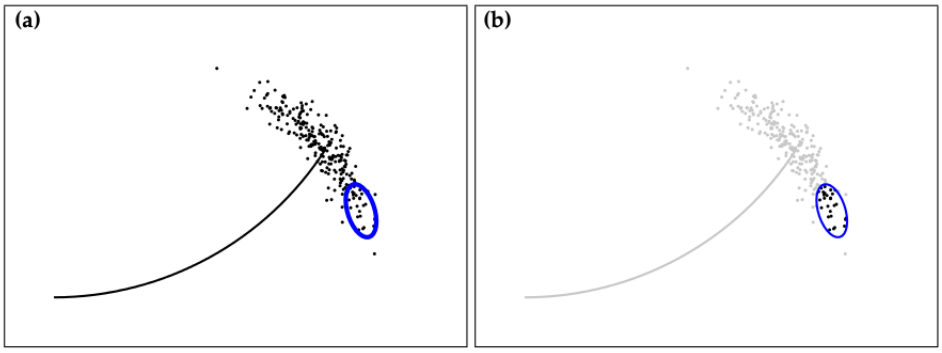


Figure 1-12: (a) Mismatch between proposal and posterior distribution, and (b) the contribution of samples after resampling [6].

motion model generates a large spectrum of samples (or particles) out of which a small subset has a high likelihood with the incoming measurements. After resampling, only few particles from the proposal distribution survive after resampling (see Figure 1-12) and this can result in loss of particle diversity or degeneracy. The state-of-the-art version of particle filter-SLAM, [12], avoids this problem by incorporating measurements into the motion model for a better prediction estimate, at the expense of intensive computation.

1-4 Research objectives

With the appropriate introduction, the thesis work will focus on a minimalistic approach to achieve Simultaneous Localization And Mapping. The main research problem for the thesis is to accomplish SLAM with an impact-based approach using minimal sensing information.

In this dissertation, I will advance the following thesis:

> *An impact-based approach to Simultaneous Localization and Mapping with minimal sensing information is proposed and implemented.*

The feature extraction in impact-based SLAM is sparse in nature and it may require a long exploration to map the environment. For example, an omnidirectional camera requires a single frame of data to map a simple rectangular environment while it may take many collisions for the impact-based approach. The thesis work is extended to use multiple robots to map the environment and merge the individual maps from the robots. This map-merging problem has a short exploration time and can also ensure a more consistent map of the environment.

1-5 Thesis contributions

In this thesis, a probabilistic formulation of the impact-based SLAM problem is addressed with a suitable robot model and an environment representation. A unicycle motion model is used to approximate the robot motion in the environment with the effect of collisions on

robot motion modelled through a Finite State Machine, and a suitable rectilinear environment representation is used for incorporating measurements from the environment.

A new efficient map representation is proposed for rectilinear environments.

The new representation models walls of a rectilinear environment as histograms and this representation is compared with the traditional point landmark representation. The new representation might share a similarity with occupancy maps since both involve discretizing the state space. However, the new map representation rasterizes only the useful part (for recording collision measurements) of the map, thereby giving it an edge over the occupancy maps in terms of memory usage and consistency. The suitability and efficiency of map representation is addressed in detail along with an implementation of the most suitable one.

The impact-based SLAM algorithm is developed using probabilistic formulation and is implemented on the practical setup.

An impact-based SLAM algorithm is developed using the robot and environment model with collision and odometry information as inputs and robot's path and map as outputs. A particle filter-SLAM is preferred over the existing state-of-the-art SLAM algorithms due to favourable properties relating to the impact-based mapping as follows. Firstly, the poor pose distribution of the robot can be easily represented through a set of particles rather than a smooth Gaussian representation. Secondly, the growth of map information can be well represented through a discrete distribution because of the availability measurement information.

A clear layout is made for the impact-based SLAM algorithm detailing the key steps for obtaining the solution. Simulation results validate the feasibility of SLAM solution and comparisons between various landmark representations are analysed. Issues such as convergence, computational effort and consistency of the solution are addressed. The impact-based SLAM algorithm is implemented on a practical setup using Sphero robot and is extended to multiple robot-SLAM.

The SLAM problem is accomplished with multiple robots as a map-merging problem for achieving a higher consistency.

The robots individually build a local map of the environment and at some point, the maps are fused together which has a higher consistency than the single robot-SLAM and is achieved with a lower exploration time.

1-6 Outline

This thesis report comprises of five chapters, including the current chapter. Chapters 2 and 3 give the necessary theoretical background for impact-based SLAM with simulation results, followed by Chapter 4 implementation of the developed algorithm. Chapter 5 summarizes the thesis with a discussion and recommendations for future work.

A detailed outline of the thesis is as follows,

Chapter 2 formulates SLAM in a probabilistic fashion suitable for impact-based SLAM. The motion model and observation model is defined for the impact-based approach along with a suitable environment representation. Various challenges to the SLAM problem are addressed.

Chapter 3 presents an impact-based SLAM algorithm using Rao-Blackwellized particle filter. Suitable illustrations and simulation results support the developed algorithm, followed by map-merging problem using multiple robots.

Chapter 4 provides the results of SLAM experimentation with Sphero robots. The robot and its sensing modalities are discussed. The experimental results for both single robot and multiple robot-SLAM are provided.

Chapter 5 summarizes the thesis with a discussion and recommendations for future work.

Chapter 2

Problem description

In this chapter, the impact-based SLAM problem is described from a probabilistic viewpoint in Section 2-2. The essentials for building an accurate SLAM solution are a robot motion model and an environment representation. Section 2-3 details the various motion models for the SLAM problem suitable for specific scenarios. Various environment representations suitable for rectilinear environments are discussed in Section 2-4 along with the proposed histogram maps. Section 2-5 addresses various challenges relevant to the impact-based SLAM problem.

2-1 Notations

In the impact-based SLAM problem, the Sphero robot collides with the walls of a given rectilinear environment randomly. The collision information, along with odometry through dead-reckoning, from the robot can infer the pose of a landmark. The odometry gives the position of collision while the collision information is used for calculation of landmark orientation. The collision is detected in Sphero when there is a sudden deceleration in either or both of the robot's coordinate axes X and Y. The wall orientation ϕ can be inferred from the collision information with respect to the robot's orientation θ by measuring the change or peak of impact along each coordinate axis. The dependence is shown below,

$$\phi - \theta = \arctan\left(\frac{a_x}{a_y}\right), \tag{2-1}$$

where a_x and a_y represent the accelerometer reading at the highest peak of impact or the power of impact along the coordinate axis X and Y respectively. Further information about collision detection can be found in Section 4-1 of Chapter 4.

At time instant t, following quantities are defined:

s_t : pose of the robot, containing position and orientation information

u_t : control input vector, containing velocity and heading, applied at time instant $t-1$ to get the pose s_t

m_i : i^{th} landmark position or pose

c_t : measurement correspondence at time t

z_t : measurement/observation from robot at time t

The time history of a variable s is denoted as $s_{0:t}$. The correspondence of a landmark to an incoming measurement is called as data association and it is crucial for any SLAM problem. This step is very important since standard estimators are prone to divergence for incorrect associations.

2-2 Probabilistic formulation

The probabilistic formulation of the SLAM describes the problem in the most certain way possible with uncertainty dealt explicitly. The uncertainty in the SLAM problem, crucial for tractability, has been modelled in few of the most common forms. For example, in the case of Extended Kalman filter, the uncertainty is modelled as a Gaussian while the particle filter models the uncertainty as a set of particles.

There are two main forms of the probabilistic SLAM problem, which are equally important in their own aspects and have their own pros and cons. One is known as the online SLAM problem, involving estimation of posterior over momentary pose along with the map. The online version is an incremental algorithm which discards past measurements and controls once they have been processed. Using the above notations, online SLAM is defined as the following posterior:

$$p(s_t, m|z_{1:t}, u_{1:t}, c_{1:t}) \tag{2-2}$$

If the set of data associations is known, the posterior becomes simple:

$$p(s_t, m|z_{1:t}, u_{1:t}) \tag{2-3}$$

A convenient way to describe a probabilistic SLAM algorithm and its structure is through a Dynamic Bayesian Network (DBN). Expressing SLAM through a DBN highlights its temporal structure as a Markov chain. A Bayesian network is a graphical model describing a stochastic process as a directed graph. The graph has one node for each variable in the process and a directed edge between two nodes models the conditional dependence between them. For further information on Bayesian networks, refer [13]. For example, the online SLAM problem as a DBN is depicted in Figure 2-1.

The other form of probabilistic SLAM is called full SLAM problem which calculates the posterior over the entire pose trajectory along with the map. The data association set is assumed to be known.

$$P(s_{1:t}, m|z_{1:t}, u_{1:t}) \tag{2-4}$$

The full SLAM problem as a DBN is depicted in Figure 2-2. The initial condition s_0 is assumed to be known in both the SLAM forms and is removed from the conditioning term for simplicity in notations.

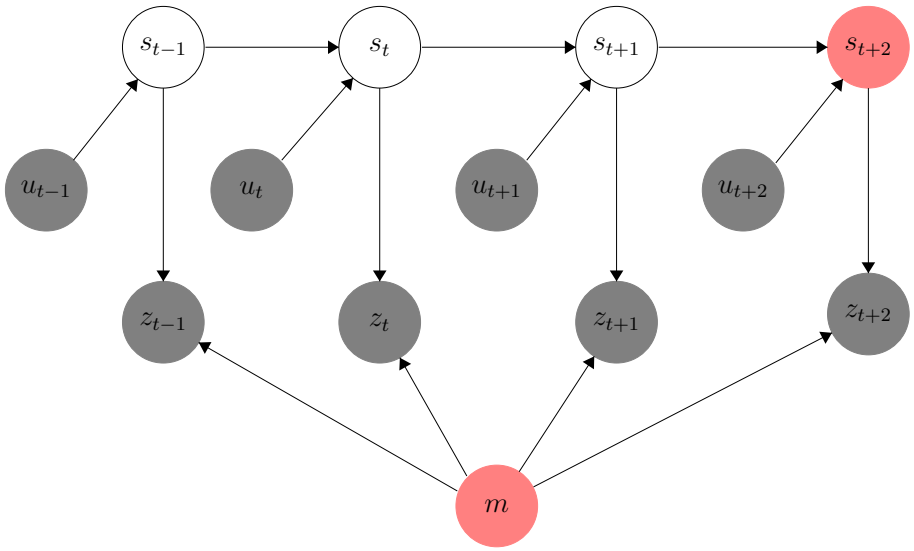


Figure 2-1: Graphical model of online SLAM problem. The online SLAM estimates a posterior over the current robot pose and map. The red shaded nodes are estimated in this problem given the gray shaded nodes.

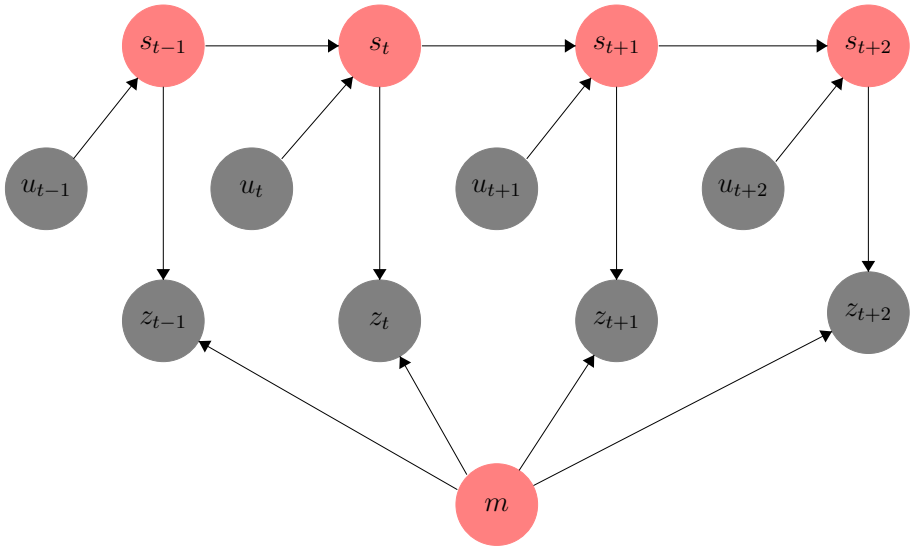


Figure 2-2: Graphical model of full SLAM problem. The full SLAM estimates a posterior over the entire robot trajectory and map. The red shaded nodes are estimated in this problem given the gray shaded nodes.

The full version of the SLAM problem involves the entire measurement and state information which results in an enormous computational complexity introduced by the huge data, while the online version is more suitable for practical applications such as navigation since the measurements are processed as they arrive. However, the full SLAM results in a more accurate and consistent estimate since it models the entire set of correlations between the measurements and robot trajectory which is neglected in the online version.

In general, a recursive solution to the SLAM problem is desirable since measurements can be processed as they arrive. The joint posterior is updated using Bayes' Theorem with the help of a state transition model and an observation model which are individually affected by control input and observation respectively.

A state transition model describes a systematic update to the pose of the robot in terms of a probability distribution of the form-

$$P(s_t|s_{t-1}, u_t). \tag{2-5}$$

The state transition model is assumed to be a Markov process in which the next state of the robot depends only upon the previous robot pose and recent control action. This structure is important for reducing the computational complexity of the SLAM problem.

The observation model describes the probability of making an observation given the robot pose and map of the environment-

$$P(z_t|s_t, m). \tag{2-6}$$

If the landmark is a new one, it has to be appended to a set of new landmarks. The landmark initialization function is defined as the inverse of observation model.

The state transition model and observation model characterizes the connectivity of the Bayesian network through a recurrent pattern as shown in Figures 2-1 and 2-2. The state transition model is represented by the edges connecting the state variable which represents the probability of state variable at that time instant. Similarly, the observation model models the probability of performing an observation given that the robot is at a location in the map. Expressing SLAM as DBN highlights its temporal structure, and this formalism is exclusively suited to describe SLAM as a filtering problem.

The recursive Bayesian update is carried out using the state transition and observation model in a two-step recursive procedure, more popularly called as Bayes filter. The Bayes filter can be derived from the SLAM posterior as follows.

Bayes filter derivation

The SLAM posterior can be rewritten using the Bayes' rule as given below.

$$p(s_t, m|z_{1:t}, u_{1:t}, c_{1:t}) = \eta \cdot p(z_t|s_t, m, z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot p(s_t, m|z_{1:t-1}, u_{1:t}, c_{1:t}) \tag{2-7}$$

The first term in the product, η, is a normalizing constant in the Bayes' rule. The fact that the current measurement z_t is solely a function of pose of the robot s_t, map m and data association n_t can be exploited to simplify the above product.

$$p(s_t, m|z_{1:t}, u_{1:t}, c_{1:t}) = \eta \cdot p(z_t|s_t, m, , c_{1:t}) \cdot p(s_t, m|z_{1:t-1}, u_{1:t}, c_{1:t}) \tag{2-8}$$

Using the theorem of total probability to condition the last term of the product on pose at time $t-1$,

$$= \eta \cdot p(z_t|s_t, m, c_{1:t}) \cdot \int p(s_t, m|s_{t-1}, z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot p(s_{t-1}|z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot ds_{t-1} \quad (2\text{-}9)$$

The leftmost term in the integral can be expanded using conditional probability theory.

$$\begin{aligned} = \eta \cdot p(z_t|s_t, m, , c_{1:t}) \cdot \int p(s_t|m, s_{t-1}, & z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot p(m|s_{t-1}, z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot \\ & \cdot p(s_{t-1}|z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot ds_{t-1} \end{aligned} \quad (2\text{-}10)$$

The first term of the above product can be simplified assuming the state propagation is a Markov process, hence s_t is only a function of s_{t-1} and u_t.

$$\begin{aligned} = \eta \cdot p(z_t|s_t, m, , c_{1:t}) \cdot \int p(s_t|s_{t-1}, u_t) \cdot p(m|s_{t-1}, z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot \\ \cdot p(s_{t-1}|z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot ds_{t-1} \end{aligned} \quad (2\text{-}11)$$

Also, the last two terms in the integral product can be combined together using Bayes' product.

$$\begin{aligned} & p(s_t, m|z_{1:t}, u_{1:t}, c_{1:t}) = \\ & \quad \eta \cdot p(z_t|s_t, m, , c_{1:t}) \cdot \int p(s_t|s_{t-1}, u_t) \cdot p(s_{t-1}, m|z_{1:t-1}, u_{1:t}, c_{1:t}) \cdot ds_{t-1} \end{aligned} \quad (2\text{-}12)$$

Since the process is causal in such a way where the current input u_t and data association c_t provide no new information about the previous state s_{t-1} or map m without the latest observation z_t, they can be dropped from the rightmost term of the integral. The result obtained turns out to be a recursive formula for computing the SLAM posterior at time t given the SLAM posterior at time $t-1$, motion model $p(s_t|s_{t-1}, u_t)$ and observation model $p(z_t|s_t, m, c_t)$.

The state transition probability assumes a Markov propagation in system states for computational efficiency. The Markov assumption states that the probability distribution of the next state depends upon the previous state and not on the sequence of states that preceded it. This requires a crucial assumption that the system is characterized by a complete state.

Definition 2.1. A complete state is defined as a characterization of a system which can be the best predictor of the future. If any state is sufficient to describe a future state with a given input, the past states can be neglected in calculations.

The error from the latter can be compensated by inflating the covariance information of the corresponding process. This assumption is quite successful in practical SLAM algorithms, especially in particle filter-SLAM algorithm where the pose error is factored out. The problem in the former case can be removed by considering a modified complete state of the system.

The accuracy of the prediction step is increased by conditioning the map state in state transition probability $p(s_t|s_{t-1}, u_t, m)$. Inclusion of map state tends to remove poor state hypotheses

in Bayes filter. A similar statement is used in the particle filter-SLAM algorithm where measurements are included in the prediction step to reduce degeneracy of particle filter. However, computation of this probability can be cumbersome and efficient approximations are used. Let us briefly derive it using Bayes' rule.

$$p(s_t|u_t, s_{t-1}, m) \cdot p(m|u_t, s_{t-1}) = p(m|s_t, u_t, s_{t-1}) \cdot p(s_t|u_t, s_{t-1}) \tag{2-13}$$

As $p(m|u_t, s_{t-1})$ can be considered a constant η, a desired approximation of $p(m|s_t, u_t, s_{t-1})$ is used as $p(m|s_t)$.

$$p(s_t|u_t, s_{t-1}, m) = \eta \cdot p(m|s_t) \cdot p(s_t|u_t, s_{t-1}) \tag{2-14}$$

A similar Bayes' rule product is obtained with a different normalization constant $\hat{\eta}$.

$$p(s_t|u_t, s_{t-1}, m) = \hat{\eta} \cdot p(s_t|m) \cdot p(s_t|u_t, s_{t-1}) \tag{2-15}$$

Inclusion of $p(s_t|m)$ can improve the accuracy of the prediction step through reduced degeneracy. For example, as the robot comes close to a wall, it is not good to represent a probability of robot being present on the wall or the other side. Hence, an inclusion of map can remove this problem. A detailed proof of the modified prediction step or proposal distribution is provided in Appendix B.

Various SLAM algorithms quite differ on the representation of these probability distributions and type of the sensors used. The following sections describe these distributions that are suitable for the impact-based SLAM. The importance and manipulation of these recursive steps will be explained in detail in future.

2-3 Robot model

The goal of a probabilistic robot model is to accurately model the specific types of uncertainty that exists in robot actuation and perception. In practice, the exact model seems to be less important than the fact that some provisions for uncertain outcomes are provided in the first place. Accurately modelling the robot is important to control the robot dynamics but in situations where certain dynamics of the robot are not excited much as part of the problem, it is not useful to model them. For the impact-based SLAM, the Sphero robot moves in straight lines between collisions and rotates around itself at the collision point. Hence, a unicycle model is sufficient to explain the behaviour of this motion and the unmodeled dynamics can be accounted as uncertainty. In case of motion planning, more accurate geometric models such as SO(3) are needed and will be considered as future work of this thesis.

2-3-1 Motion model

There are two specific probabilistic motion models for the Sphero mobile robot. Both the models are somewhat complimentary in the type of motion information being processed. The first model, called velocity motion model, assumes that the motion data specifies the velocity and heading commands given to the robot. The second model, called odometry motion model, assumes that the motion information is provided through the odometry. The latter is more

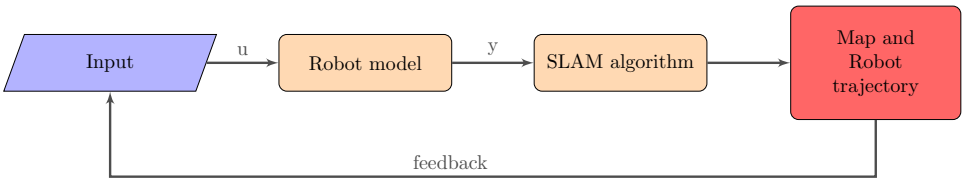


Figure 2-3: Velocity model for SLAM with planning; u denotes control input and y denotes the measurement including the odometry. The feedback enables planning in SLAM.

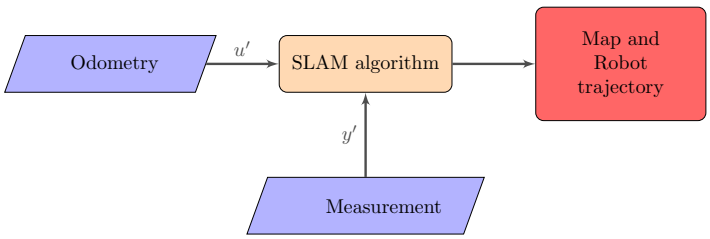


Figure 2-4: Odometry model for SLAM; u' denotes odometry input and y' denotes the collision measurement

accurate than the former for a simple reason that error in the unmodelled robot kinematics are not accounted and the information is available post-the-fact. The velocity motion model is most suitable for autonomous robots as it incorporates the robot motion model which is essential for motion planning.

The two models used in SLAM algorithms are shown in Figures 2-3 and 2-4.

Velocity motion model

A sampling based approach is used to compute the state transition probability from a given state $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$ and control input $[\text{speed}, \text{head}]$. The time from $t-1$ to t is denoted dt and during this period, the control input is constant. The samples are randomly generated (**gauss**) from a Gaussian distribution either using Central Limit Theorem or a more efficient Box-Muller technique. After the sampling step, the pose is propagated through a unicycle robot model to get a new pose. The computation of the state transition probability is shown in Algorithm 1.

Algorithm 1 Sampling from velocity motion model

1: **Input**: $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$, [speed, head], dt
2: **procedure** SAMPLING
3: speed = **gauss**(speed, $\alpha_1\sqrt{speed}$)
4: head = **gauss**(head, $\alpha_2\sqrt{head}$)
5: **procedure** PROPAGATION
6: angle = $(head + \phi_{t-1})\%(2\pi)$
7: $x_t = x_{t-1} + \text{speed} \cdot dt \cdot \textbf{cos}(\text{angle})$
8: $y_t = y_{t-1} + \text{speed} \cdot dt \cdot \textbf{sin}(\text{angle})$
9: $\phi_t = \phi_{t-1} + \text{speed} \cdot dt \cdot \textbf{sin}(\text{head}/L)$
10: **return** $s_t = [x_t, y_t, \phi_t]$

Figure 2-5: New pose $s_t = [x_t, y_t, \phi_t]$ is sampled from old pose $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$ with a sample time dt using controls [speed, head] from a robot of wheel base L.

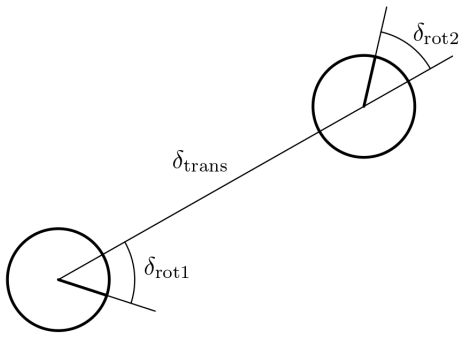


Figure 2-6: The robot motion is approximated by a sequence of rotation δ_{rot1}, translation δ_{trans} and rotation δ_{rot2} [6].

Odometry motion model

Odometry measurements are obtained from the IMU of Sphero robot through integration of accelerometer data and gyroscopic data (called as dead-reckoning). A similar sampling approach is adopted for the odometry based approach but without the robot model. The input to the sampling algorithm is previous robot pose estimate $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$ and the odometry reading, $u = [[\bar{x}_{t-1}, \bar{y}_{t-1}, \bar{\phi}_{t-1}], [\bar{x}_t, \bar{y}_t, \bar{\phi}_t]]$. The propagation is more simple than the previous where the propagation step is described with separate translational and rotational movements as shown in Figure 2-6. The propagation approach is suitable for very small movements since larger time steps might result in robot jumping over features which cannot be explained through the robot's motion model. The computation of state transition probability is shown in Algorithm 2.

Algorithm 2 Sampling from odometry motion model

Input: $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$, $[[\bar{x}_{t-1}, \bar{y}_{t-1}, \bar{\phi}_{t-1}], [\bar{x}_t, \bar{y}_t, \bar{\phi}_t]]$
procedure CONTROLS
  $\delta_{\text{rot1}} = \textbf{atan2}(\bar{y}_t - \bar{y}_{t-1}, \bar{x}_t - \bar{x}_{t-1}) - \bar{\phi}_{t-1}$
  $\delta_{\text{trans}} = \sqrt{(\bar{y}_t - \bar{y}_{t-1})^2 + (\bar{x}_t - \bar{x}_{t-1})^2}$
  $\delta_{\text{rot2}} = \bar{\phi}_t - \bar{\phi}_{t-1} - \delta_{\text{rot1}}$
procedure SAMPLING
  $\hat{\delta}_{\text{rot1}} = \textbf{gauss}(\delta_{\text{rot1}}, \alpha_1\delta_{\text{rot1}} + \alpha_2\delta_{\text{trans}})$
  $\hat{\delta}_{\text{trans}} = \textbf{gauss}(\delta_{\text{trans}}, \alpha_3\delta_{\text{trans}} + \alpha_4(\delta_{\text{rot1}} + \delta_{\text{rot2}}))$
  $\hat{\delta}_{\text{rot2}} = \textbf{gauss}(\delta_{\text{rot2}}, \alpha_5\delta_{\text{rot2}} + \alpha_6\delta_{\text{trans}})$
procedure PROPAGATION
  $x_t = x_{t-1} + \hat{\delta}_{\text{trans}} \cdot \textbf{cos}(\phi_{t-1} + \hat{\delta}_{\text{rot1}})$
  $y_t = y_{t-1} + \hat{\delta}_{\text{trans}} \cdot \textbf{sin}(\phi_{t-1} + \hat{\delta}_{\text{rot1}})$
  $\phi_t = \phi_{t-1} + \hat{\delta}_{\text{rot1}} + \hat{\delta}_{\text{rot2}}$
  return $s_t = [x_t, y_t, \phi_t]$

Figure 2-7: New pose $[x_t, y_t, \phi_t]$ is sampled from old pose $s_{t-1} = [x_{t-1}, y_{t-1}, \phi_{t-1}]$ through a set of translation δ_{trans} and rotations δ_{rot1} and δ_{rot2}. The last input is the consecutive odometry measurement from time $t-1$ to t.

2-3-2 Observation model

The measurements available from the Sphero robot are odometry and collision measurements. The odometry from the IMU provides the robot pose through dead-reckoning and the correlation between the robot position from odometry and landmark position is straightforward since for the impact-based SLAM, the robot is present at the absolute location of collision (except for an offset which can be corrected in the map). The nonlinear relation between the collision information $\frac{a_x}{a_y}$, robot orientation θ and landmark orientation ϕ is shown below,

$$\phi - \theta = \arctan\left(\frac{a_x}{a_y}\right)$$

The resulting observation model of a landmark l is shown,

$$y_l = \begin{bmatrix} x_r \\ y_r \\ c_r \end{bmatrix} = \begin{bmatrix} x_l \\ y_l \\ \tan(\phi - \theta), \end{bmatrix}$$

where (x_l, y_l) denotes the landmark position, (x_r, y_r) denotes the robot odometry and c_r denotes the collision information. Note that the robot orientation θ is also a part of the measurement. A probabilistic version can be obtained from the observation model much similar to the sampling approach in motion model.

2-4 Environment representation

A suitable environment representation is needed to build a map of the environment and use the map to estimate the robot trajectory. The accuracy of a map representation is directly correlated with the accuracy of the SLAM solution, and care is taken to represent the environment in the best possible way.

In the case of impact-based SLAM, the given environment is an indoor world which comprises of rectilinear walls. The orthogonality assumption in walls is crucial for orientation correction for the robot. In addition, equal corridor width in the environment is a common assumption in indoor environments for correcting the position of the robot [14]. Additional assumptions such as boundary length are taken for building a map to ensure consistency if needed.

The most suitable representations of an indoor environment are point landmarks, line representations and occupancy maps. The first is more simple than the other while the last involves more book-keeping. The later representation is more useful for the case of planning, and its variants are in use for the Google autonomous car. The line representation is suitable for a SLAM problem with a range-and-bearing sensor. The suitability of a map representation is deeply correlated with the data association (refer 2-5-3) used for SLAM. Suitable environment representations for the impact-based SLAM problem are presented below.

2-4-1 Point landmarks

The point landmarks are described using pose information, or in other words, position and orientation (though mathematically, it is not appropriate for a point to be described using orientation). For the case of impact-based SLAM, the collision information has to be encoded in such a way to initialize and update corresponding landmarks for each and every collision.

The point landmark representation is the most preferred representation in literature for analysis. The map is defined as a set of point landmarks where each landmark m_i is defined by its pose and possible signatures.

$$m = \begin{bmatrix} m_1 & m_2 \cdots & m_N \end{bmatrix} \tag{2-16}$$

The simplest description of a point landmark is by its position $m_i = \begin{bmatrix} x_i & y_i \end{bmatrix}^T$. Additional components such as landmark signatures (e.g. color, texture) can be added to improve the description.

For a 3-dimensional pose description of a point landmark, the uncertainty associated is also 3-dimensional in nature and can be modelled as a multivariate Gaussian.

$$\mathcal{N}\left(\begin{bmatrix} x_i \\ y_i \\ \phi_i \end{bmatrix}, \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{x\phi} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{y\phi} \\ \sigma_{x\phi} & \sigma_{y\phi} & \sigma_{\phi\phi} \end{bmatrix} \right) \tag{2-17}$$

For the case of impact-based SLAM problem, the use of point landmark is insufficient to completely represent the environment since the environment is rather more descriptive than point landmark representation. In addition, even if SLAM is carried out for a simple corridor

environment, new landmarks are created for every collision and it is very rare that the robot touches the same landmark again.

The above situation can however be avoided with two possible solutions-

1. The uncertainty associated with each landmark can be increased such that close collisions could be regarded as a single landmark. However, this can result in overestimation or a highly conservative estimate, and result in an inaccurate solution. This situation is exemplified in Example 2.1.

2. The spatial information of the wall can be taken into account for clubbing collisions on the same wall. This technique involves taking only uncertainty along the orthogonal direction since orthogonal direction of uncertainty can be reduced from collision updates. However, the associated result was found to be poor since the environment information is not completely encoded, as exemplified in Example 2.2.

A more suitable representation is required for a better description of the indoor environment as point landmarks were no more suitable. The data association results tend to fail a lot with these representations and hence, more complex representations are needed to better suit the environment and the algorithm. A new representation is required to suit a rectilinear environment for impact-based SLAM as well as address the issues of point landmark representation.

Example 2.1. *Consider the Sphero robot mapping a corridor environment. The robot moves in a straight line between consecutive collision and the uncertainty associated with each collision is modelled as a Gaussian. The collision information is encoded as a point landmark and the associated algorithm is detailed.*

In standard EKF-SLAM, landmark pose is a component of mean of the multivariate Gaussian and associated uncertainty is maintained as sub-covariance of the Gaussian. A landmark is created from a measurement that does not correspond (nearest-neighbour gating) to any old landmark and updated if the corresponding landmark has a higher likelihood. Note that the association of point landmarks to measurements in this example holds for any data association algorithm in literature.

This approach is found to be less suitable for point landmarks since new landmarks are created for every collision within walls and there are no EKF updates. Hence, the associated uncertainty grows unbounded and there is no convergence in the solution. This can be illustrated in Figure 2-8.

In this example, five landmarks are created rather than two landmarks. The state vector of EKF-SLAM as a result becomes

$$x = \begin{bmatrix} x_r \\ y_r \\ \phi_r \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \end{bmatrix} \tag{2-18}$$

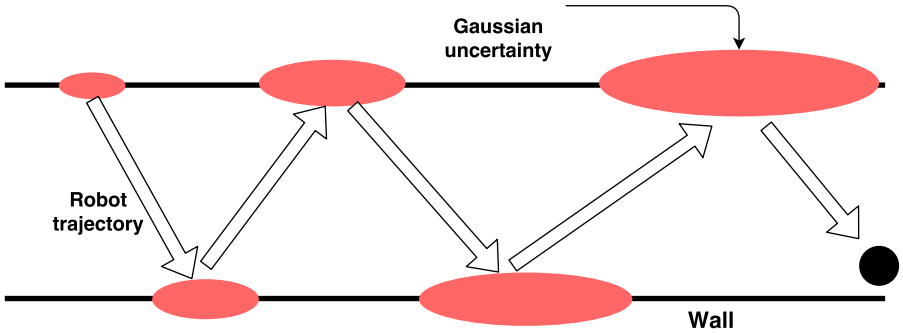


Figure 2-8: New landmarks are created for collisions on the same wall. The robot coordinate frame is same as map coordinate frame.

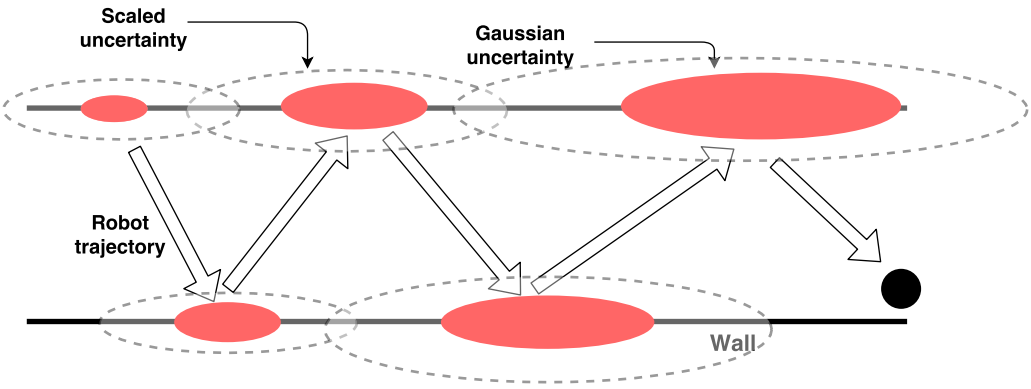


Figure 2-9: Scaled uncertainty to improve data association over the same wall

The size of state is found to increase linearly in the number of collisions and the uncertainty correspondingly grows unbounded.

A suitable approach to overcome this issue is to go for scaled or inflated representation of uncertainty [15] such that close-by landmarks of the same wall can have lower Mahalonobis distance. The scaled representation is illustrated in Figure 2-9 .

The resulting EKF-SLAM state turns out to be

$$x = \begin{bmatrix} x_r \\ y_r \\ \phi_r \\ m_1 \\ m_2 \end{bmatrix} \tag{2-19}$$

Consistency of map via this approach might be achieved by inflating the covariance [15] but it is not clear how to quantify adequate inflation. Even though the representation may appear

simple, this technique may result in an unbounded growth in covariance with a non-converging solution. This is due to partiality in the uncertainty updates along the unit normal direction of the wall, which is usual in rectilinear environments. □

Example 2.2. *Consider a similar situation to Example 2.1 where the robot collides along the walls of a corridor. Spurious landmark creation were avoided by inflating the covariance matrix to ensure a consistent solution for the map. However, the map structure can be taken into account to reduce the overall scaling of the uncertainty representation.*

A reduced state representation of the landmark state is chosen for data association. Let X-axis denote the orthogonal direction to the wall and Y-axis denote the axis along the wall. As the Y-axis correspond to the same landmark along the wall direction, collisions on the same wall (Y-axis) also correspond to the same landmark. Hence, the X-axis coordinate and orientation (reduced state) can be used for data association since the X-axis does not contribute to any useful information for data association. This introduces a bounded covariance for this reduced state representation. The uncertainty along the wall axis (Y-axis) can be reduced when a joint covariance representation of all the landmarks are maintained or a Covariance Intersection (CI) update is used between neighbouring orthogonal landmarks for incorporating orthogonal measurement information (Refer Section 4-2).

In addition, information about the geometric structure of the environment can be incorporated. Constraints such as equal corridor width, orthogonality in wall orientation and boundary length can be incorporated into EKF as zero-uncertainty measurements [16]. Addition of these measurements can reduce the growth of covariance and ensure consistency of the SLAM solution.

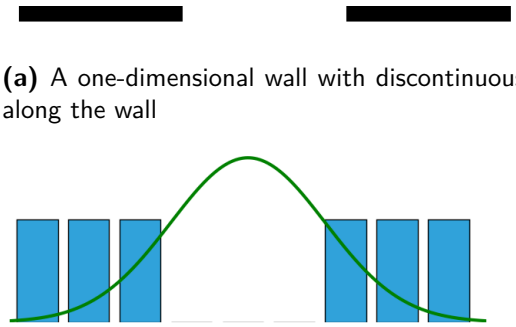
The corridor width w in an usual indoor environment is assumed to be constant [14]. This structure can be exploited to reduce the uncertainty along the orthogonal direction to the wall. For example, the two walls, each parametrized as a hyperplane $a_i^T x = b_i$ where x can be same as Equation 2-19, can model the equal width constraint as

$$y = w = a_1^T \cdot x - a_2^T \cdot x = b_1 - b_2 \tag{2-20}$$

Similarly, the orthogonal assumption and boundary length can be incorporated in a similar fashion. This technique can avoid the spurious landmark creation but at the expense of additional computation complexity through a joint covariance or constraint incorporation. □

The last approach still has an important disadvantage (Example 2.1) which is poor representation of the environment. The information of wall mass is not represented anywhere in point landmarks. For example, the spatial information (wall mass) of a discontinuous wall (as illustrated in Figure 2-10) cannot be properly modelled as a Gaussian and it requires a more elaborate representation such as occupancy map or particles. Moreover, if distant collisions of the same wall are clubbed as a single landmark, combinatorial possibilities can open up whether they correspond to the same landmark or not during the pruning step (door openings might be present in between two collisions on the same wall).

The suitability of point landmarks for impact-based SLAM is also shown through simulations in the forthcoming chapter.



(a) A one-dimensional wall with discontinuous mass along the wall

(b) Histogram and Gaussian representation of the wall mass

Figure 2-10: Spatial representation of wall mass information for a simple 1-D environment

2-4-2 Occupancy maps

A more popular and descriptive representation of a map than a point landmark is an occupancy grid where the entire map of an environment is rasterized into small cells. Each cell encodes the occupancy information and as a result, processing of information and estimation [6] becomes more simple. For example, a cell (i, j) is described by an occupancy state x as,

$$x(i,j) = \begin{cases} 1 & \text{if cell is occupied} \\ 0 & \text{if cell is free} \\ -1 & \text{if cell is unknown} \end{cases}$$

The occupancy map representation does not include any information about landmark signature and this map representation has been used extensively for planning and navigation since most of the planning algorithms favour a grid-structured map.

Occupancy maps can represent any arbitrary distribution at the expense of higher memory requirements. Impact-based SLAM in literature ([17], [18]) use an occupancy grid representation to account for the poor uncertainty distribution in impact-based mapping and SLAM.

This thesis strikes a mid-level between a point landmark representation and occupancy map representation, by exploiting the advantages of both the approaches and aiding each other at shortcomings. Without further delay, the forthcoming section will detail the histogram representation of a map.

2-4-3 Histogram maps

The thesis proposes a hybrid approach to an environment representation, much suitable for rectilinear environments. A landmark is represented using a point landmark description as well as with a histogram description. The landmark will turn out to be more descriptive than point landmarks and at the same time, the representation requires less of book-keeping compared to occupancy maps. Histogram representation of a 1D map has been used in literature [6] for pedantic purposes and not for a real-world implementation since occupancy maps turn out to be more descriptive.

Before getting deeper into histogram maps, a histogram is first defined.

Definition 2.2. A histogram is a bar graph, illustrating the mapping $\mathcal{F}$ between finite domain $\mathcal{X}$ and its associated range $\mathcal{Y} = \mathcal{F}(\mathcal{X})$. A histogram can also represent a probability distribution where the domain represents a finite set of events and the range represents the associated probability of an event.

Taking a more specific example (refer Figure 2-10) such as a wall in an indoor environment for impact-based SLAM, the domain represents discretized wall location along the wall. The associated range describes the probability of mass information along the wall. To make it more clear, if X-axis describes the thickness of wall and Y-axis describes the wall length, the histogram is defined along the Y-axis and encodes information about the presence of wall.

As said in the previous section, histogram maps strike a mid-level between point landmark representation and occupancy grid representation. For an indoor environment, each wall is described as a point oriented landmark where its pose is determined from the initial collision which lead to its inception. As a new landmark gets created, a new state describing its pose is augmented to the system state. In addition to the new state, a histogram with a finite support is also initialized for the corresponding landmark. The length of support is taken from the prior knowledge of the size of the environment. If the size of the environment is not perfectly known, an approximate bound is taken for the size and pruned later for a consistent map.

Figure 2-10 illustrates the histogram representation of a discontinuous wall. The histogram can represent any discontinuous, non-Gaussian distribution and shares similarities with occupancy maps. This representation is aptly suited for impact-based SLAM since the collision information can be encoded into the histogram by propagating it through discrete Bayes filter while landmark pose is propagated using EKF similar to previous point representation. The discrete Bayes filter is a simple extension of continuous Bayes filter (refer Equation 1-2) where the integral gets converted to a finite sum. Spurious landmark creation is avoided using the approach followed in Example 2.2. The propagation of a histogram using discrete Bayes filter for an impact-based SLAM problem is shown in Example 2.3.

Example 2.3. *Consider a wall in an one-dimensional world which contains an opening (open door). This can be illustrated in Figure 2-11. A robot equipped with touch sensors, can move back and forth along the wall and make contacts at certain points. In addition, the position of a robot is given by a noisy odometry. A simple assumption is made here that there is no measurement noise associated with the touch sensors.*

Figure 2-11: One-dimensional wall for impact-based mapping

The histogram is propagated as a probability distribution using discrete Bayes filter with a sequence of incoming measurements. The histogram represents the probability of occupancy along the wall and hence the initial distribution is assumed to be uniform. Two possible implementations of a collision update are Gaussian update and triangular update where the probability is computed at discrete points using these distributions.

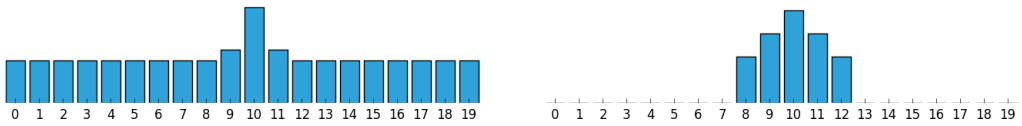


Figure 2-12: Propagation of histogram for an incoming measurement. The left histogram is propagated as a probability distribution whereas the right is a simple histogram update (not a probability distribution).

It is not necessary that histogram has to be declared as a probability distribution and use discrete Bayes' theorem for mapping the wall. A simple histogram update can be used where each cell contains a probability of occupancy (much similar to occupancy maps). Both the approaches of histogram representation have the same computational efficiency. Moreover, maintaining a collinear histogram also improves the consistency of the estimate along the wall. Figure 2-12 gives the difference between the two histogram update techniques for a single collision.

Figure 2-13 give a list of histogram updates along the wall, with collisions occurring according to the cell sequence $\{10, 4, 12, 13, 1, 5, 9, 6, 7, 19\}$. A triangular collision update is used here for simplicity. The corresponding map of the wall is shown for each collision based on the simple histogram update (for a good illustration of map formation).

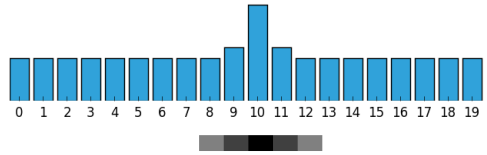
In the case the robot passes through an open door and returns back, a simple negative update is added. This update is illustrated in Figure 2-14. □

2-5 Challenges

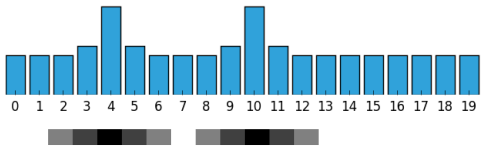
There are few challenges playing a key role to the success of a SLAM solution which limit the applicability based on robot sensors and mapping environment. Among the following challenges, lack of observability is a key fundamental limitation to state estimation in a generic SLAM. In a usual SLAM estimation, observability is presumed (with knowledge of few absolute locations) and hence it has to be made sure that the impact-based SLAM satisfies the observability property as well. Other crucial factors are data association, computational effort and consistency of estimates. The computational issues related to state-of-the-art SLAM estimation have been addressed in Section 1-3-1 of Chapter 1.

2-5-1 Observability

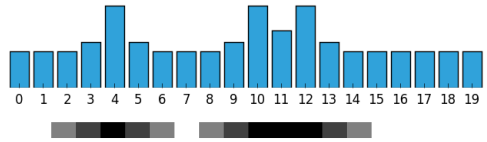
Observability is a key property to be satisfied for any successful state estimation. SLAM is a nonlinear-coupled problem and hence simple linear analysis tools fail to explain the observability of SLAM. It is apparent from process model (Equation 2-23) is nonlinear and coupled. This also holds true for the odometric motion model where the nonlinear and coupled nature is introduced through the collision measurement equation (Equation 2-24).



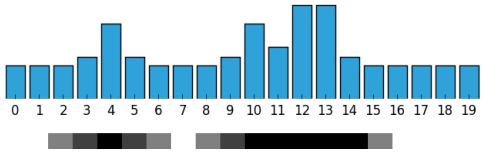
(a) First collision at 10 and the resulting map



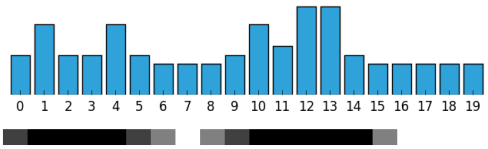
(b) Next collision at 4 and the resulting map



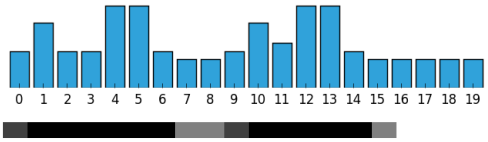
(c) Next collision at 12 and the resulting map



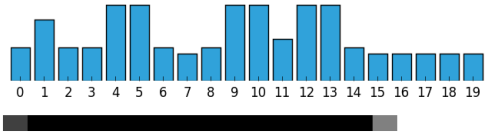
(d) Next collision at 13 and the resulting map



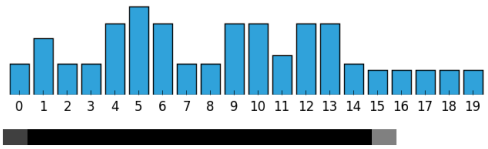
(e) Next collision at 1 and the resulting map



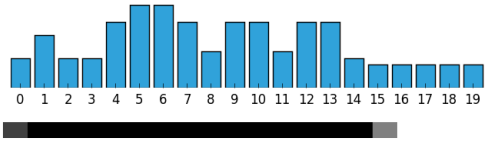
(f) Next collision at 5 and the resulting map



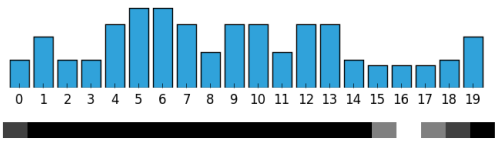
(g) Next collision at 9 and the resulting map



(h) Next collision at 6 and the resulting map



(i) Next collision at 7 and the resulting map



(j) Last collision at 19 and the resulting map

Figure 2-13: Histogram propagation for a sequence of collisions

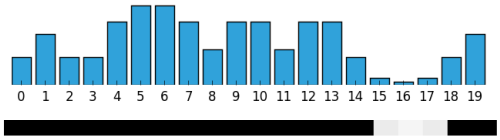


Figure 2-14: Negative update of histogram at 15 and the resulting map

To determine the observability criterion for a general nonlinear coupled system, let us consider a smooth nonlinear system,

$$\dot{x} = f(x) + \sum_{j=1}^{m} g_j(x) \cdot u_j, \qquad u = col(u_1, \ldots, u_m) \in \mathcal{R}^m$$
$$y_i = h_i(x), \qquad i \in p$$

where $h = col(h_1, \ldots, h_p) : \mathcal{R}^n \rightarrow \mathcal{R}^p$ is a smooth output map or vector field of the system.

The concept of observability is linked to *indistinguishability* of a state since the effect of control inputs play a role in nonlinear observability. Before defining observability, the definition of *indistinguishability* is first laid out.

Remark 2.1. *Two states x_1 and x_2 are* V*-distinguishable if there exists an input which drives system trajectories from both x_1 and x_2 remain in* V *such that the output functions are distinguishable $h(x_1) \neq h(x_2)$. Otherwise, the states x_1 and x_2 are* V*-indistinguishable.*

The system is said to be locally observable if it can instantaneously distinguish a state from its neighbours. A rank condition test is used to determine the local observability of a given system, as shown below,

$$span(\mathcal{O}) = span(\Big[dh_1(x_0), \ldots, dh_p(x_0), dL_{v_k} L_{v_{k-1}} \cdots L_{v_1} h_j(x_0)\Big]) = n, \tag{2-21}$$

where $v_i, i \in k$ is a vector field in the set $\{f, g_1, \ldots, g_m\}$ and $L_{v_i} h_j$ is the Lie derivative of h_j along the vector field v_i.

For a world-centric impact-based SLAM problem, the system state can be represented as,

$$x = \begin{bmatrix} x_r & y_r & \psi_r & x_l & y_l & \psi_l \end{bmatrix}, \tag{2-22}$$

where first three variables represent the robot pose (position x and y, orientation ψ) while the latter represents the landmark pose.

The representation included a single feature and can be generalized for observing as many features as possible. Using the process (unicycle) model,

$$\begin{aligned} \dot{X} &= \frac{d}{dt} \begin{bmatrix} x_r & y_r & \psi_r & x_l & y_l & \psi_l \end{bmatrix}^T \\ &= \begin{bmatrix} \cos\psi_r & \sin\psi_r & \tan(\alpha)/L & 0 & 0 & 0 \end{bmatrix}^T v, \end{aligned} \tag{2-23}$$

where v and α represent the robot velocity and heading angle respectively. L denotes the wheel base length.

For the impact-based SLAM problem, the odometry as measurement gives the position and orientation of the robot. A nice feature about impact-based SLAM is that at the instant of collision, the absolute location of landmark is the same as the absolute location of the robot. The measurement function as a result has a diagonal structure which makes the Jacobian computation and observability analysis easier.

The observation of a landmark given the robot and observed landmark pose is,

$$y_l = \begin{bmatrix} x_r \\ y_r \\ c_r \end{bmatrix} = \begin{bmatrix} x_l \\ y_l \\ \tan(\psi_l - \psi_r), \end{bmatrix} \tag{2-24}$$

where y_l is the expected sensor measurement of the l^{th} landmark. The measurement c_r is a analytical expression obtained through the geometry of robot impact against a rectilinear wall.

By taking zero-order Lie derivatives, we can get the rank condition satisfied assuming the initial condition of the system or initial robot pose is known.

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -s^2 & 0 & 0 & s^2 \\ & & \vdots & & & \end{bmatrix} \tag{2-25}$$

The rank condition is satisfied with $rank(\mathcal{O}) = 6$. The result can be generalized for observation of as many landmarks as possible. s^2 represents $\sec^2(\psi_l - \psi_r)$ which is the derivative of measurement equation.

However, there is a downside to the impact based mapping for SLAM. At the instant of collision, the robot is touching the wall which makes the assumption of a smooth pose distribution invalid. The resulting pose distribution can only be represented by a set of particles which makes particle filter most suitable for localization. This is because the landmark (covariance) representation has to be consistent with the current robot pose (covariance) representation at the point of collision and this has to hold vice-versa. This fundamental representation of probability distribution must hold true for propagation of pose and landmark estimates to ensure consistency in estimation. Figure 2-15 illustrates the representation of probability distribution for impact-based approach to mapping.

2-5-2 Correlation and Consistency

The correlation through robot pose error has been considered a crucial factor in the SLAM problem since it has contributed to the tractability of the problem. However, the problem of

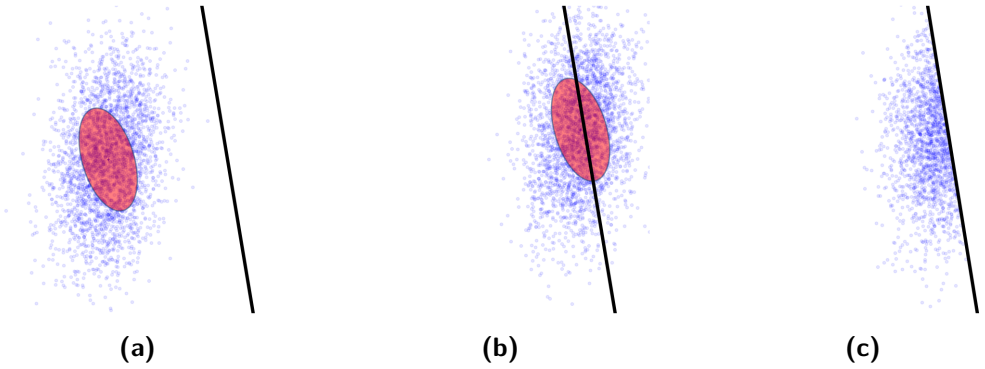


Figure 2-15: (a) Gaussian and sampled representation of robot pose uncertainty moving towards a wall, (b) Gaussian and sampled representation of robot pose uncertainty for colliding against the wall, (c) Sampled representation of pose uncertainty taking wall constraints into account

introducing uncertainty as a bridging factor between localization and mapping has their own ups and downs.

The main advantage, as mentioned, is the tractability in solving the SLAM problem since it can be used to improvise the robot pose estimate through reobservation of old landmarks. The robot tries to correct its pose and simultaneously improves the pose of all the previously observed landmarks due to the common correlation in the observations. The solutions dealing with explicit correlations are shown to be successful in practical implementations [19]. The representation of correlations through uncertainty information has led to the popularity of the probabilistic methods in SLAM.

However, there is a downside to the use of robot uncertainty (process noise) in the problem as the bridging factor. The effect of using uncertainty to influence the observability in the problem has a dig at the consistency of the estimation. To improve the convergence of the solution to the problem, the correlation have to improve with time, or said in other words, higher the correlation, better is the SLAM. The problem is that increase in uncertainty, if not properly modelled in SLAM, can be really problematic since it can lead to optimistic estimates of the uncertain state variables which cannot be compensated unlike in usual tracking and navigation problems.

Before getting into the problems of consistency due to the structure and assumptions in SLAM, the definition [20] is first laid out.

Definition 2.3. An estimator is said to be consistent if the estimated state converges to the true value and remains thereon unless an external disturbance acts on it. If e_n is the error in the estimate, the true covariance or variation in the actual system state can be written as $E(e_n^T e_n)$. Σ_n is the estimated covariance or estimated variation of error in the system.

Mathematically, an unbiased estimator is consistent if and only if it can capture the true variation in error.

$$\Sigma_n = E(e_n^T e_n) \tag{2-26}$$

However, this condition of consistency is difficult to establish due to the stochastic scenario of a system. To enforce consistency, conservativeness is introduced so that we can bound the

true variation of error at all time instances. Mathematically,

$$\Sigma_n - E(e_n^T e_n) \geq 0 \tag{2-27}$$

A standard solution to ensure consistency is to inflate the observation covariance by padding it with sufficiently large values. This process is called as "covariance inflation" or injection of stabilizing noise into the system [15]. There is however a problem associated with this technique. Due to the inflation of noise covariance, the assumed uncertainties in certain directions of state space increases and as a result, the steady state covariance can grow unbounded.

Consistency in the SLAM scenario can be discussed based on the estimator or optimization technique to be implemented for the SLAM problem. The question of consistency can raise from several issues related to the SLAM problem such as data association errors, modelling errors, transformation errors.

The effect of data association is highly sensitive and a single incorrect association will lead to inconsistency in the solution. The first issue has been addressed a lot in the literature [21] and is detailed for the impact-based SLAM problem in the next section. The effect of inconsistency due to probability transformation errors have been studied and suitable solutions are developed in [22]. The inconsistency issues relevant to EKF-SLAM and particle filter-SLAM are discussed below.

EKF-SLAM

The structure and properties of the Extended Kalman Filter-SLAM, as discussed previously, will be revisited to discuss about source and effects of inconsistency. The inconsistency in EKF-SLAM arises from linearization error in observation and inverse observation models.

To ensure consistency, the following condition [20] has to be satisfied,

$$\nabla h_s - \nabla h_{m_i} \nabla g_s = 0 \tag{2-28}$$

where ∇h_s, ∇h_{m_i} represent the Jacobian of the measurement function with respect to robot state and landmark estimate respectively. ∇g_s denotes the Jacobian of inverse measurement function with respect to the state. Usually, this function is never satisfied due to linearization errors and as a result, the estimates turn inconsistent. As the relationship depends upon the nonlinear system models and noisy state estimates, the linearization error is difficult to compute and cannot be compensated through inflation of error covariance.

The effect of linearization errors can be reduced through local map fusion which uses the robot's local coordinate frame to update the map [23]. This approach has a lower error in the state estimates which in turn reduces the effect of inconsistency. The use of multiple robots in impact-based SLAM ensures a higher consistency in the local map updates which can be later merged. Other approaches such as observability constrained estimation [10] also ensures a more consistent state estimation at every time step.

Particle filter-SLAM

An advantage about the use of particle filters is elimination of inconsistency due to probability transformation errors since the particle set can be propagated through any nonlinear model.

However, there are two other key issues related to inconsistency in particle filter-SLAM. The first issue is inconsistency through modelling errors and the second i degeneracy in the estimation process.

The particle filter-SLAM problem exploits the Rao-Blackwellized property where landmark estimation problems are conditional independent given the robot trajectory. This property is based on on the assumption that the environment is unstructured, or in other words, the environment contains randomly placed landmarks. The only correlation factor is the robot pose error in the SLAM problem which is conditioned to make the landmark estimation independent. However, in the case of a rectilinear environment, the landmarks are correlated through the geometric symmetry in the environment such as orthogonal walls. Hence, a straightforward implementation of Rao-Blackwellized particle filter to SLAM problem results in inconsistent estimation.

The effect of cross-correlations cannot be ignored since they are crucial for ensuring consistency in the SLAM estimation. The lack of proper correlations leads to a poor covariance estimate which leads to inconsistency (refer Remark 4.1). The cross-correlations in the environment can be effectively maintained using a joint covariance matrix (a single EKF for all the landmarks) for every particle in the particle set. This approach might lead to computational issues since every measurement update requires a computation complexity of $\mathcal{O}(M \cdot N^2)$ which is cumbersome.

A more computationally efficient approach is to conservatively incorporate the conditionally independent landmarks (through Rao-Blackwellization) with the orthogonality and partial uncertainty constraints through Covariance Intersection (CI) technique [24]. This approach ensures a consistent estimation in a conservative fashion at the expense of optimal performance (rate of map convergence). With this approach, the computational efficiency of Rao-Blackwellization is still maintained.

For the case of degeneracy, the particle filter-SLAM produces optimistic uncertain estimates in the long term, [9]. In general, particle filters provide a consistent recursive estimation through resampling in situations where a system exponentially forgets its past estimate errors. The problem in particle filter-SLAM is that the past pose errors are never forgotten since they are recorded in the map estimates through correlations and each time a resampling is performed, the particle that is not selected gets thrown out with an entire map hypothesis. Hence, the particle filter degenerates a good knowledge of map hypotheses and the most likely particle (with largest importance weight) duplicates by replacing other unlikely particles in the resampling step.

As a result, there is a variance reduction in the particle set which can result in an underestimated covariance. The algorithm starts to get optimistic about the uncertainty in the trajectory of the robot and the landmark locations. Every time a particle is lost due to resampling, an entire map hypothesis is lost and there is a depletion of historical information. Hence, the overall map statistics degrade and the algorithm finds it difficult to close larger loops. Moreover with the loss of past information, data association errors cannot be cor-

rected based on the future measurements which brings brittleness in data association similar to standard EKF-SLAM.

In the short term, particle filter-SLAM might produce consistent results given a sufficient number of particles. To avoid this problem, adaptive resampling techniques are adopted to ensure long term consistency in the solution. In addition, impact-based SLAM have sparse measurements with frequent loop closures (in corridors) which might result in degeneracy. This can be avoided through the inclusion of measurements in the prediction step, resulting in a more consistent estimate with less number of particles. Section 3-2-2 and 3-2-6 detail the modified prediction step and and adaptive resampling techniques for the impact-based SLAM problem.

2-5-3 Data Association

Data association determines the mapping from the measurements to the landmarks and it is one among the essentials for achieving a successful SLAM solution. The association is quite sensitive to the uncertainty of robot involved in SLAM and few incorrect associations can lead to inconsistency in the SLAM estimation. Over the past years, this issue has been dealt seriously and robust data association techniques were developed to ensure consistency.

The data association algorithm depends upon the characteristics of landmarks and availability of measurements. Two of the most popular approaches for data association are Maximum Likelihood (ML) and Joint Compatibility Branch and Bound (JCBB) [7], former being the simplest and vulnerable one. The later approach is most useful for a SLAM where batch measurements are available for processing at each time instant. Apart from the two approaches, Multiple Hypotheses data association [25] has been the most accurate of all but with a computational burden of maintaining all the possible associations of landmarks with measurements.

For impact-based SLAM using particle filter, each particle maintains a copy of the map and hence, data association decisions can be made on a per-particle basis. This leads to a robust multiple hypothesis association. The particles are not only a representation of uncertainty of state estimate but are also individual hypothesis of the map. Hence, Multiple Hypotheses is ingrained in the particle filter-SLAM formulation.

Among the data association algorithms, the most suitable one is Maximum Likelihood (ML) association where each particle performs a ML association for the incoming measurements.

Remark 2.2. *In data association, following assumptions have to be considered for an analytical representation of landmark likelihood to a measurement.*

1. *The incoming measurements are normally distributed, i.e., the measurement noise is Gaussian*
2. *Knowledge of the Gaussian parameters, mean and standard deviation, are available a-priori*

The data association is computed using likelihood function, also known as χ^2 test. The maximum likelihood problem is posed as shown below,

$$d_l = \arg\max_j \frac{1}{(2\pi)^{n/2} \cdot \sqrt{S_j}} \exp\left(-\frac{1}{2} e_{ij}^T \cdot S_j^{-1} \cdot e_{ij}\right), \tag{2-29}$$

where d_l is the l^{th} data association for measurement i and landmark j. The variable e_{ij} is the innovation vector for the pair $\{i, j\}$ pair, S_j is the innovation covariance matrix and n is defined as the dimension of the innovation vector.

Remark 2.3. *The solution can also be computed through log-likelihood for numerical stability. This approach is more suitable for data associations over multiple measurements or smoothing process in the estimation process.*

In the case of multiple measurements or smoothing process, the measurements are considered as a vector and are assumed to be independent. Hence, the likelihood is defined as joint probability, which under the measurement independence assumption, factor into a product of marginal probabilities and the magnitude of the likelihood can be quite small, often very close to zero. With a large number of observations, this value can approach the machine zero of the computing device used, leading to numerical problems. Log-transforming all these probabilities will convert these small numbers into negative numbers thus eliminating numerical instability. The solution however still remains to be the same due to the increase monotonicity of the log-function.

Taking logarithm over the likelihood, the problem is rewritten as shown below.

$$d_l = \arg\max_j \left(-\frac{1}{2} e_{ij}^T \cdot S_j^{-1} \cdot e_{ij} + \ln \left(\frac{1}{(2\pi)^{n/2} \cdot \sqrt{S_j}} \right) \right) \tag{2-30}$$

An equivalent normalized-minimization problem can also be posed as follows,

$$d_l = \arg\min_j \left(e_{ij}^T \cdot S_j^{-1} \cdot e_{ij} + \ln(|S_j|) \right). \tag{2-31}$$

Summary In this chapter, a probabilistic formulation of the impact-based SLAM problem is laid out from the perspective of online estimation. Two robot motion models are described using a unicycle model framework for the SLAM problem. Suitable environment representations are studied for the impact-based SLAM along with the proposed histogram map. Various challenges such as observability, correlations, consistency and data association are addressed from the viewpoint of impact-based SLAM.

Chapter 3

SLAM solution

This chapter focusses exclusively on the solution to the impact-based SLAM problem. A modified version of the particle filter-SLAM algorithm will be developed to suit rectilinear environments, and the problem is extended to multiple robots for map-merging. Simulation results support the developed algorithm for both single robot and multiple robot-SLAM.

3-1 Factored representation

In particle filter-SLAM, the uncertainty of robot pose is represented by a particle set (see Figure 2-15) rather than a smooth parametric Gaussian as in EKF-SLAM. This representation has great advantages of representing any arbitrary multi-modal distribution, and they can be easily propagated through any probabilistic robot motion models.

However, there are downsides to implementing particle filter for SLAM due to a key reason. The number of particles scales exponentially with the dimension of the state space which restrict the use of particle filters to low-dimensional estimation and tracking problems. A suitable representation of particle filter towards the SLAM problem would be essential for an accurate SLAM solution. As discussed in Section 1-3-2, the conditional independence property of the mapping problem given the robot trajectory is a key structure which can be exploited. This structural property in SLAM or any probabilistic problem that factorizes a joint distribution to conditional distributions is *Rao-Blackwellization*.

Appendix A will provide a detailed proof of the conditional independence property in SLAM posterior and the exact nature of the factorization. Under this structure, cross-correlations between the landmarks introduced by the robot path uncertainty need not be maintained, and in the case of structured environments, explicit constraints are incorporated through CI technique for ensuring consistency in estimation. The SLAM algorithms in future will utilize the conditional independence property for efficient computation of the posterior.

3-2 Particle filter-SLAM

Using the factored representation of the SLAM posterior, the SLAM solution can be obtained with linear time complexity in the number of landmarks, and the accuracy of obtained solution is proportional to the size of the particles. The particle filter-SLAM algorithm is simple and intuitive and a detailed algorithm is provided at each and every step.

Before delving into an algorithm, inclusion of data association into the SLAM posterior has to be discussed. The previous examples and illustrations assumed that data association is known, which is not the usual in SLAM. The general SLAM posterior is,

$$p(x_{1:t}, m, c_{1:t}|z_{1:t}, u_{1:t}), \tag{3-1}$$

where the it can be factorized through Bayes' product rule,

$$p(x_{1:t}, m, c_{1:t}|z_{1:t}, u_{1:t}) = p(x_{1:t}, m|z_{1:t}, u_{1:t}, c_{1:t}) \cdot p(c_{1:t}|z_{1:t}, u_{1:t}). \tag{3-2}$$

The first term of the product is same as the old SLAM posterior and the second term correspond to the data association conditioned upon the observations. The vector $c_{1:t}$ is a vector of correspondences where each component makes a correspondence between a feature and a measurement.

Remark 3.1. *The second term of the product in Equation 3-2 corresponds to the data association, conditioned on the observations but not on the robot path. This discrete prior distribution about landmarks is not available initially and has to be engineered according to the sensors and operating environment.*

The FastSLAM algorithm maintains the (first term of the product) posterior distribution as a set of samples where each particle in the sample set corresponds to the state vector. According to the conditioned distribution, the map has to be conditioned on the robot trajectory and the length of each particle will grow over time. However, the update equations in FastSLAM, as in Section 1-3-2, do not depend upon the entire trajectory but only upon the most recent pose. Hence, each particle needs to contain the most recent pose and a map of the environment.

Each particle in the particle set $Y_t = \left\{Y_t^{[1]}, Y_t^{[2]}, \ldots, Y_t^{[M]}\right\}$ of size M takes the form as in Equation 1-13,

$$Y_t^{[k]} = \left\langle x_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle \tag{3-3}$$

The k^{th} particle contains the recent robot pose and a corresponding set of landmarks of size N where landmark i is described by mean $\mu_{i,t}^{[k]}$ and covariance $\Sigma_{i,t}^{[k]}$ at time t.

From the particle set, the particles are propagated or sampled from the probabilistic motion model and the corresponding step is called as the Prediction step. The particles are then updated using incoming set of measurements and resampled if necessary. A simple flowchart explaining the process involved in the FastSLAM algorithm is illustrated in Figure 3-1.

The following sections will detail the individual steps involved in the algorithm.

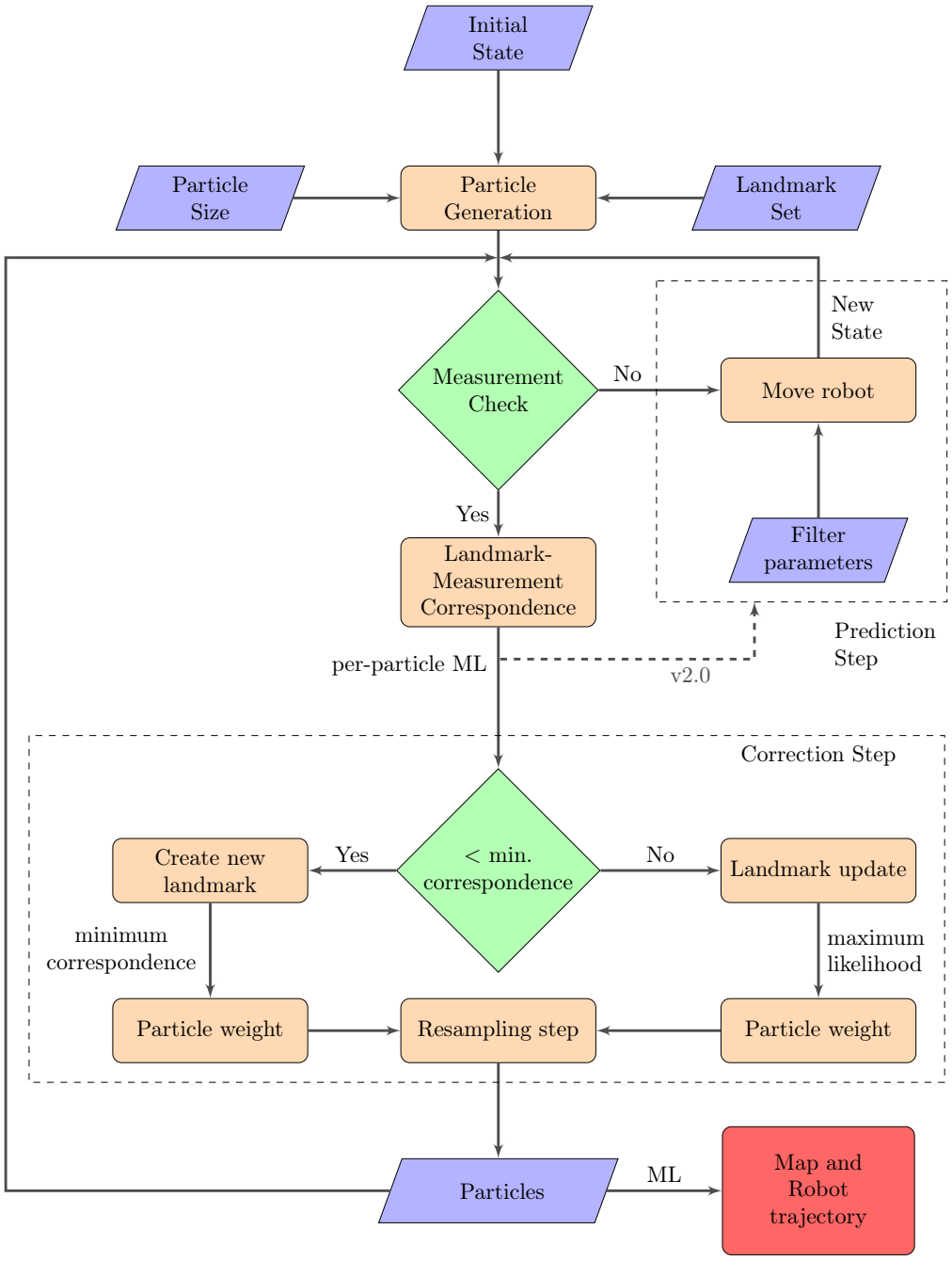


Figure 3-1: Particle filter-SLAM algorithm

3-2-1 Particle generation

Initially, a particle set of given size has to be created from a known distribution. The particles are either sampled from a Gaussian distribution whose mean correspond to the initial state of the robot and covariance corresponding the pose error or they are sampled from a prior particle distribution. In the SLAM scenario, the initial uncertainty is taken to be zero with no prior knowledge of landmarks.

$$Y_t^{[k]} = \langle x_0 \rangle \tag{3-4}$$

As all the particles have the same initial state, the individual maps will also have the origin as the initial state.

Remark 3.2. *The initial uncertainty in the position estimates (both robot and landmark) can also be non-zero, which ultimately converges to a stable solution. This can improve the initial diversity of the estimates and thereby ensuring a more consistent solution at the expense of time to converge to an accurate map.*

3-2-2 Prediction step

The new state estimate is sampled from the old state using the probabilistic motion model and controls. The two motion models suitable to the SLAM scenario, along with the algorithm, are detailed in Section 2-3.

The propagation of particles through the motion model has a linear time complexity $\mathcal{O}(M)$ and the new state estimate from each particle constitute to form a temporary particle set called as proposal distribution. This distribution takes the following form asymptotically,

$$p(s_t|u_t, s_{t-1}) \tag{3-5}$$

As the measurement likelihood is sparse or discrete with respect to the pose space in impact-based SLAM, the SLAM solution can be approximated to a Gaussian using Markov assumption for the measurement model. The current measurement depends upon the current robot pose and not on the previous set of poses. This assumption holds valid only since a complete state (refer Definition 2.1) is defined for the impact-based SLAM problem. Hence, the prediction step has to be repeated till a measurement is made available. The robot moves between the walls and a measurement is available only during a wall collision. As a result, the uncertainty in robot pose increases till a measurement is available.

The resulting motion model can be quite noisy and as said in the previous Chapter, this might result in a degeneracy problem. This arises when the robot motion is more noisy than the measurements. In the case of impact-based SLAM, the motion noise gets accumulated till a single measurement is made. Less particles will now be able to explain the new measurement and hence, particle diversity and consistency are lost during resampling(see Section 2-5-2 for more details).

A favourable solution to the degeneracy problem is to modify the prediction step to include the measurements such that the proposal distribution is generated in accordance to incoming measurements [12]. The modified proposal distribution can be written as,

$$p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) \tag{3-6}$$

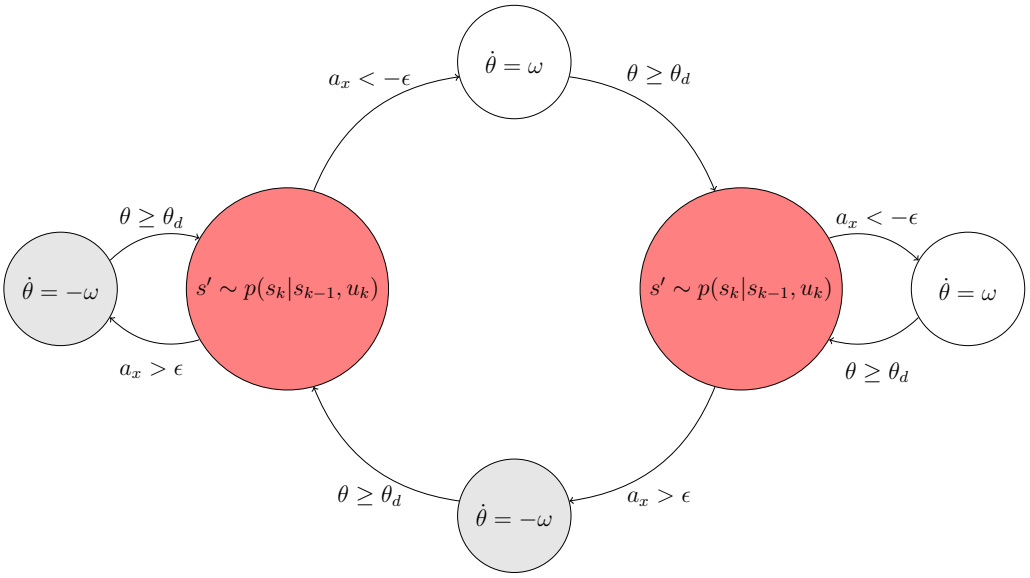


Figure 3-2: Finite State Machine model of the robot motion between collisions. The red shaded nodes indicate the motion model of the robot. The gray and white shaded nodes indicate a specific control action on orientation corresponding to the sign of a_x. θ_d indicates the final desired angle through the function $T_i(\cdot)$ as in Equations 3-8 and 3-9.

In short, the variance propagation of measurements get included into the prediction step to obtain a better proposal distribution that can explain the incoming measurement. Hence, the proposal distribution remains closely same as the posterior distribution and particle diversity is retained. Appendix B gives a detailed proof of the modified proposal distribution.

For impact-based SLAM, the robot changes its heading direction (orientation in yaw axis) after every collision, which is encoded through a Finite State Machine (FSM) as shown in Figure 3-2. The FSM is modelled taking into account the exact nature of motion of robot between collisions. In forward motion, the robot can collide with the wall either on the left side or on the right side as shown in Figure 3-3. This can restrict the size of FSM to two translational motion states (red shaded nodes) as in Figure 3-2 where the left side represents the heading of robot towards a wall from left side and the other from right side. The rotation states describe the change in robot's orientation after every collision.

The underlying motion model used for the FSM in prediction step is the same unicycle model in Section 2-3 as follows,

$$s' \sim p(s_k|s_{k-1}, u_k) \tag{3-7}$$

The change in robot's orientation depends upon the collision of the robot with the wall from the left or right hand side. With a coordinate frame is assigned to the robot, the forward motion of the robot is Y-axis and to the right is X-axis. As the collision detection is achieved through changes in the accelerometer values, the left-handed collision of the robot with a wall has negative acceleration on the X-axis while the right-handed collision of the robot with a wall has positive acceleration on the X-axis. This can be described through a change in

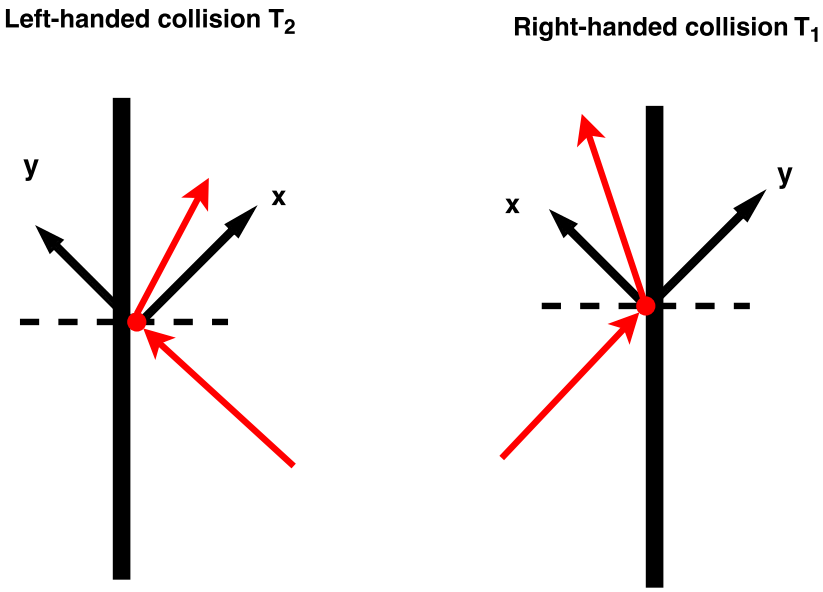


Figure 3-3: Approach of robot towards a rectilinear wall restricts to two possible cases, left-handed collision and right-handed collision. The black arrows depict the local coordinate frame of the robot before collision and the red arrows depicts the robot motion. The ratio of change in impact force along the individual axis is used as collision measurement.

orientation for the robot where the left-handed collision model is written as,

$$T_1(\theta, a_x) = \theta - \left(\pi - c \cdot \arctan\left(\frac{a_x}{a_y}\right)\right), \tag{3-8}$$

and the right-handed collision model as,

$$T_2(\theta, a_x) = \theta + \left(\pi - c \cdot \arctan\left(\frac{a_x}{a_y}\right)\right). \tag{3-9}$$

The constant c is the reflection constant which varies according to the requirement of desired heading after a collision. For a reflective collision i.e., angle of incidence is same as angle of reflection, the value of c is 2. The initial orientation θ in yaw axis of the robot is obtained from the IMU. Further details of Sphero robot motion after collision is detailed in Section 4-2.

Remark 3.3. *The false positives in collision detection may arise from the positive acceleration in the forward direction. This can be avoided by considering only the positive acceleration hyperplane in collision detection, i.e.,* $a_y > 0$.

3-2-3 Data Association

The data association for the impact-based SLAM performs a landmark-measurement likelihood computation on a per-particle basis. Three key advantages with the per-particle data association are as follows,

- Multi-hypothesis data association is the robust form of data association till date.
- Consistent landmark updates can be carried out on a per-particle basis since landmarks in a particle do not depend upon robot pose uncertainty as robot pose error is factored out.
- Data association decisions can be revisited by removing particles which cannot explain incoming measurements and resampled with a particle which is consistent with new measurements.

The association is sensitive to the source of uncertainty arising from the motion or measurement model. An increase in measurement noise can lead to a wrong association of a measurement to a landmark. This is because as the measurement uncertainty increases, the distribution of measurements from nearby landmarks start to substantially overlap. As a result, some landmarks get wrongly associated which might lead to a underestimation (spurious reduction) of covariance. This problem will have a minor impact on the SLAM solution, provided landmarks are not close by. However, the noise from the motion model will have a more drastic impact since higher uncertainty in the robot pose leads to a wide distribution of particles and the resulting per-particle data association will be quite different. This situation occurs in the impact-based SLAM problem where the motion noise gets accumulated between the sparse measurements. This problem in association can be reduced using the modified proposal distribution as mentioned in the previous section.

Algorithm 3 Per-particle data association

```
1: Input: z_t, R_t , Y_t and m_{1:N}
2: for each particle ∈ Y_t do
3:     l ← [ ]
4:     for each landmark ∈ m_{1:N} do
5:         compute expected measurement
6:         if geometric constraint not satisfied then
7:             l.append(0)
8:         if landmark in robot's view then
9:             compute likelihood
10:            l.append(likelihood)
11:        else if check similar features then
12:            create feature if not present in landmark list
13:            compute likelihood
14:            l.append(likelihood)
15:        else
16:            l.append(0)
17:    return correspondence ← max(l)
```

Figure 3-4: Per-particle data association is carried out using an incoming measurement z_t, measurement covariance R_t, particle set Y_t and landmark set $m_{1:N}$.

Using the per-particle data association, the worst-case computational complexity involved in computing the likelihood is proportional to $\mathcal{O}(M \cdot n^3)$, where M is the size of particle set and n is the dimension of output vector. This can be computationally intensive which can be reduced on an average case through a pruning process. The process prunes the landmark with orientations which are not in the field of view for the robot, which also contributes to finite nature of FSM in a rectilinear environment. In addition, geometric constraints can be added to improve the efficiency of association. One such geometric constraint is the feature pose of the same landmark. In the robot's view, feature orientation of 0 degree always come to the right of a feature with an orientation of 180 degrees with respect to the robot's frame.

In computing the measurement likelihood, correspondence was computed only from the set of point landmarks, and histogram information is not used. Similar to occupancy maps, histogram maps can be used for localizing the robot using the occupancy information. However, a sufficient amount of collision information along the wall is necessary to influence the localization. The use of histogram information (as a probability distribution) in data association will be more promising in situations where the rectilinear landmarks are more randomly placed.

3-2-4 Landmark update

The data association step might provides a landmark identity once its correspondence with a measurement satisfies the compatibility test and it is found to be more likely than other landmarks. As the data association step is carried out on a per-particle basis, the corresponding

landmark update is also carried similarly. The rest of the landmark set remain unchanged.

The landmark update stage is a part of the mapping problem, described through the posterior $p(m_l|s_t, u_{1:t}, z_{1:t}, c_{1:t})$. Expanding the term using Bayes' product,

$$p(m_l|s_t, u_{1:t}, z_{1:t}, c_{1:t}) = \eta\, p(z_t|m_l, s_t, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(m_l|s_t, u_{1:t}, z_{1:t-1}, c_{1:t}) \tag{3-10}$$

The first probabilistic term of the above product corresponds to the measurement model and is approximated as a Gaussian using EKF linearization. The second term is the landmark estimate conditioned on the particle pose which is equal to the mean and covariance estimates of the landmark maintained by a particle $\mathcal{N}\left(\mu^{[k]}_{l,t-1}, \Sigma^{[k]}_{l,t-1}\right)$.

EKF uses a linear approximation of the measurement model $h(\cdot)$ which is obtained through a first order Taylor series expansion. According to the conditioned measurement distribution, each landmark update in a particle is carried out over a fixed robot pose. This property favours computation since the Jacobian over robot pose $G_{s^{[k]}_t}$ in an EKF update is unnecessary. As a result, the Jacobian over the associated landmark $G_{m_{l,t}}$ is only calculated in the Taylor expansion.

$$\hat{z}_t = h(s^{[k]}_t, \mu^{[k]}_{l,t-1}) \tag{3-11}$$

$$G_{m_{l,t}} = \nabla_{m_l} h(s_t = s^{[k]}_t, m_l = \mu^{[k]}_{l,t-1}) \tag{3-12}$$

$$h(s_t, m_{l,t}) \approx \hat{z}_t + G_{m_l}(m_{l,t} - \mu^{[k]}_{l,t-1}) \tag{3-13}$$

The mean of the Gaussian is approximated till the first order Taylor expansion and the covariance is assumed to be known initially. The derivative of the measurement function is defined everywhere except in cases where robot and wall orientation are the same (robot collides parallel to the wall) at the point of collision. This situation is not possible in impact-based SLAM using Sphero. The parametrization of a Gaussian measurement can be written as,

$$\mathcal{N}(\hat{z}_t + G_{m_l}(m_l - \mu^{[k]}_{l,t}), R_t) \tag{3-14}$$

The mean and covariance of the associated landmark can now be computed using the standard EKF update equations.

$$S_{l,t} = G_{m_l} \Sigma^{[k]}_{m_{l,t-1}} G^T_{m_l} + R_t \tag{3-15}$$

$$K_t = \Sigma_{m_{l,t-1}} G^T_{m_{l,t}} S^{-1}_{l,t} \tag{3-16}$$

$$\mu^{[k]}_{l,t} = \mu^{[k]}_{l,t-1} + K_t\,(z_t - \hat{z}_t) \tag{3-17}$$

$$\Sigma^{[k]}_{l,t} = (I - K_t G_{m_l})\, \Sigma^{[k]}_{l,t-1} \tag{3-18}$$

The term $S_{l,t}$ and K_t represent the innovation covariance of landmark l and Kalman gain at time t respectively.

Negative evidence

A new histogram is created once a new landmark is detected. This histogram is maintained either as a probability distribution or as a simple histogram for encoding the collision information. In the case of a probability distribution, the histogram is defined as a uniform distribution, based on the prior assumption that the distribution of mass information is unknown. The probability mass propagates into the regions where collisions were detected and a multimodal distribution is generated as seen in Figure 2-12.

With the histogram representation, negative information can be incorporated into the map. As the robot passes through the histogram axis in the map, a negative Gaussian information is added to the histogram as shown in Figure 2-14. This information helps in pruning the histogram support. If the environment size is known prior, the histogram support can be made finite according to the environment size.

3-2-5 Landmark initialization

The landmark is initialized each time a new measurement is unable to explain all the landmarks in the landmark set or when the landmark set is empty. The data association step in the previous section assumed that all the landmarks were available prior.

The landmarks are initialized using the inverse measurement model, provided the measurement function (see Equation 2-24) is invertible. The inverse is ill-defined in a SLAM problem when a given measurement is unable to constrain the landmark state and multiple measurements are required (as in range-only or bearing-only SLAM). However, the measurement model of impact-based SLAM is invertible since the landmark pose at time t is calculated through the conditioned robot pose, odometry and collision information at time t.

$$\mu_{l,t}^{[k]} = \begin{bmatrix} x_l \\ y_l \\ \psi_l \end{bmatrix} = \begin{bmatrix} x_r^{[k]} \\ y_r^{[k]} \\ \psi_r^{[k]} + \arctan\left(\dfrac{a_x}{a_y}\right) \end{bmatrix} \tag{3-19}$$

The landmark pose m_l is described as a vector of landmark position (x_l, y_l) and orientation ψ_l. Similarly, the robot pose for a particle k is described as a vector of robot position $(x_r^{[k]}, y_r^{[k]})$ and orientation $\psi_r^{[k]}$. a_x and a_y denote the rate of change in accelerometer data along the X and Y axes respectively.

Remark 3.4. *The offset between the position of landmark and robot at the instant of collision varies by a constant factor. This factor should be taken into account for the resulting output map of SLAM.*

As the measurement is obtained from a nonlinear model, a first order Taylor expansion (as in Equation 3-13) can reveal the Gaussian nature of the noise where the difference in the approximation is the error e_t,

$$e_t = z_t - \hat{z}_t - G_{m_l}(m_l - \mu_{l,t}^{[k]}) \tag{3-20}$$

and the Gaussian as,

$$\mathcal{N} := \frac{1}{\sqrt{|2\pi R_t|}} \exp\left(-\frac{1}{2} e_t^T R_t^{-1} e_t\right) \tag{3-21}$$

The landmark covariance is obtained through the covariance bound of the χ^2 distribution (see Definition 3.1). The measurement covariance propagation through the SLAM state space is obtained by projecting the measurement covariance into the landmark state space, which is equivalent to calculating the second differential of Equation 3-22. As the landmark is defined using inverse of the measurement model, the landmark covariance is the inverse of the measurement covariance propagation.

Definition 3.1. The χ^2 distribution is the negative exponent of the Gaussian $\mathcal{N}$ as shown below,

$$\chi^2 := \left(z_t - \hat{z}_t - G_{m_l}(m_l - \mu_{l,t}^{[k]})\right)^T R_t^{-1} \left(z_t - \hat{z}_t - G_{m_l}(m_l - \mu_{l,t}^{[k]})\right). \tag{3-22}$$

As a result, the mean $\mu_{l,t}^{[k]}$ is obtained through the inverse of measurement model (Equation 3-19) and the covariance $\Sigma_{l,t}^{[k]}$ is obtained from the second differential of the Equation 3-22 and taking the inverse since an invertible measurement is used to define a new landmark.

$$\mu_{l,t}^{[k]} = h^{-1}\left(s_t^{[k]}, z_t\right) \tag{3-23}$$

$$\Sigma_{l,t}^{[k]} = \left(G_{m_l}^T R_t^{-1} G_{m_l}\right)^{-1} \tag{3-24}$$

As the measurement model has a diagonal symmetry where each state depends upon only few variables, the Jacobian with respect to landmark states will be diagonal as well, which gives way to a computationally efficient approach to a initialize a new landmark.

A faster initialization (on average time efficiency) procedure can be performed for covariance computation for all the landmarks by assuming a spherical initial covariance. The magnitude of this uncertainty has to be computed by conducting trials on various values of magnitude through an eigenvalue decomposition (average computational complexity is $\mathcal{O}(n^3)$) to recover the largest magnitude. This approach is however numerically unstable since the uncertainty involved in the update process is high.

In addition to the landmark initialization using mean and covariance which is common to histogram and point landmark representation, a histogram has to be created with a finite support especially for a histogram map. The length of the support depends upon the boundary length or can be set arbitrarily and pruned later. Example 3.1 gives a detailed representation of defining a new landmark with a histogram.

Example 3.1. *Consider a simple 2-D rectilinear environment as shown in Figure 3-5, with a robot doing impact-based SLAM. The walls are numbered from 1 to 6 in red color and landmarks are initialized using EKFs as in Equations 3-23 and 3-24.*

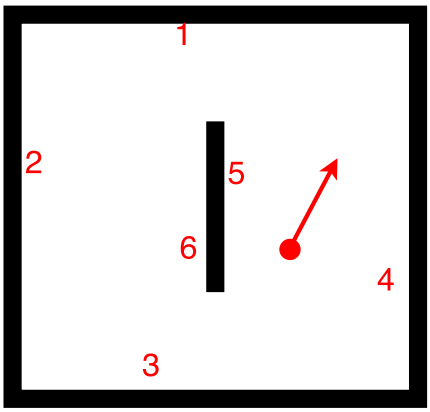


Figure 3-5: A simple rectilinear world

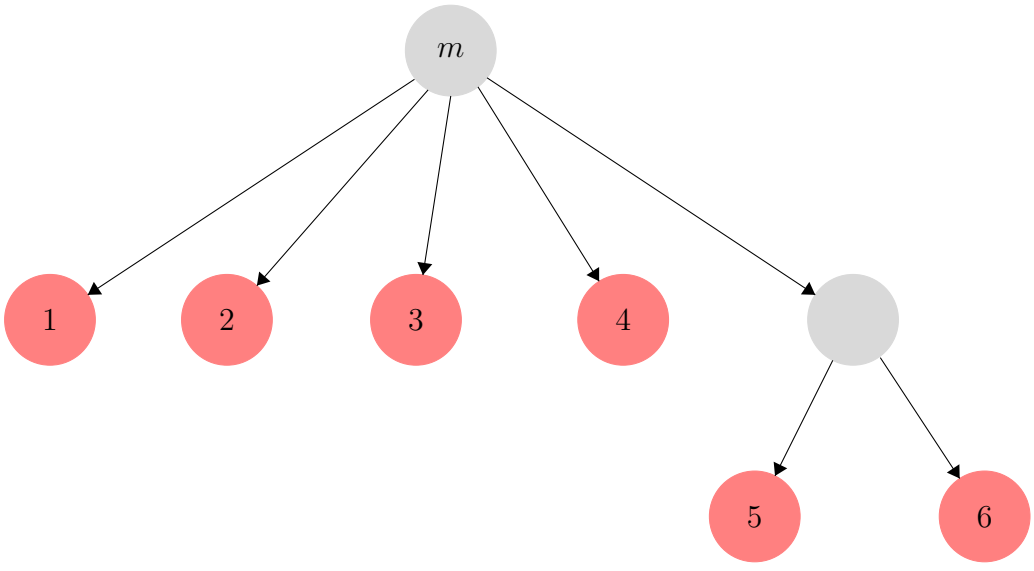


Figure 3-6: A tree data structure for landmark representation in Example 3.1

The landmarks are defined and propagated using a tree representation which is efficient in access and storage. The map is created as a root node or a pseudo-node to the tree. The individual landmarks are appended as a leaf node to the tree. A simple geometrical structure is exploited to reduce the number of leaf nodes in the tree. Walls 1 to 4 can be regarded as a landmark and a histogram has to be created for each of them to encode the collision information. However, for walls 5 and 6 represent the same wall back-to-back and a single landmark pose is sufficient to maintain. Hence, walls 5 and 6 are regarded as features of the same landmark and a single histogram is sufficient. The representation is illustrated in Figure 3-6. □

3-2-6 Importance resampling

After the particles are updated using an incoming measurement based on the maximum correspondence likelihood, the SLAM algorithm is not yet done. The mapping part of SLAM has been completed for the entire particle set where the localization (robot pose in particle) is found to influence the map. According to the SLAM definition, the robot must able to build a map using its pose as well as localize itself within the built map, or in simple words in relevance to the context, the localization and mapping must improvise each other. Hence, the mapping part of SLAM has to influence the robot pose or equivalently, the particle set to be distributed according to the posterior belief.

An improvised set of particles are drawn for the next iteration through resampling. Each particle is weighted in proportion to the likelihood of the measurements using the sensor model (refer Section 1-3-2),

$$p(z_t|s_t, m), \tag{3-25}$$

where this likelihood computation is done in the data association step (refer Appendix B for importance weight calculation for the case of modified proposal distribution). This technique in particle filters is called Sequential Importance Resampling (SIR). As the likelihood of measurements gets higher, the particles can explain the incoming measurements well, and with a higher probability they are drawn for the next iteration. Hence, the particles with an accurate robot pose are more likely to retain in the distribution and localization is achieved through the mapping process with the help of resampling.

After few resampling steps, the particles might lose diversity and there might be a single particle that got duplicated several times. This is a crucial problem in particle filters called as the degeneracy problem. This has a huge impact in the consistency of SLAM solution (as explained in Section 2-5-2) since particles cannot revisit their past data association history. Usually, consistency is ensured by resampling particles that can explain the measurements which holds true only for systems which exhibit forgetting of past estimation errors. This is not the case for SLAM since the past pose error plays a key role through landmark-pose correlations for obtaining a consistent solution. The inconsistency arises when a particle lost through resampling, the corresponding pose and a entire map hypothesis gets lost. The past pose information gets lost along with the corresponding map estimate and might result in poor loop closures. Greater diversity in a particle set ensures the SLAM algorithm to revisit their past associations and prune them (by assigning low weights) if necessary.

In such a case, an adaptive resampling technique [26] can be adopted where a criterion can be established about when to perform a resampling step. The criterion has to find the *effective*

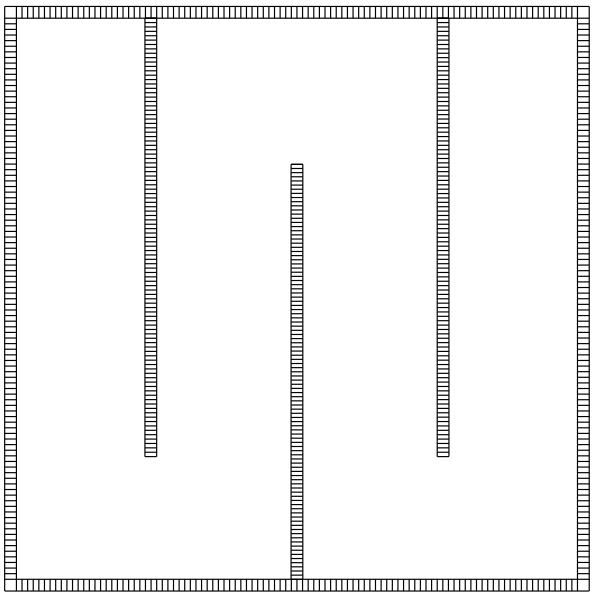


Figure 3-7: Sample world for the simulation environment

number of particles N_{eff} which estimates how well the current set of particles represents the posterior distribution. This criterion can be computed as follows,

$$N_{\text{eff}} = \frac{1}{\sum_{k=1}^{N} \left(w^{[k]}\right)^2} \tag{3-26}$$

The intuition behind the use of N_{eff} is simple as follows. If the particles are drawn from the true posterior, the variance in the weights achieves a minimum and grows as the approximation of true posterior deviates. This measure of dispersion can inform when a resampling step can be performed. As N_{eff} falls below a suitable threshold (usually around $N/2$), the particles are resampled. As a result, particle diversity is ensured by maintaining a good set of particles or hypothesis.

3-3 Simulation Results

This section supports the SLAM algorithm developed in the previous section with simulation results in the form of graphs and output map. A two dimensional world, as shown in Figure 3-7, is considered for the simulation with a ball robot modelled using unicycle-FSM model.

The odometry is obtained using the accelerometer and gyroscope through dead-reckoning. In the ideal case, the position information provided by the robot should be same as the robot's actual position, provided there are no slip between the robot and the ground. However, the odometry information is subject to significant drift which has to be accounted in the simulation scenario. Since the collision information is also obtained from the robot's IMU, it is also subject to drift. Figure 3-8 shows the drift in the odometry from the true position, which is considered in the simulation. The drift is obtained through addition of Gaussian

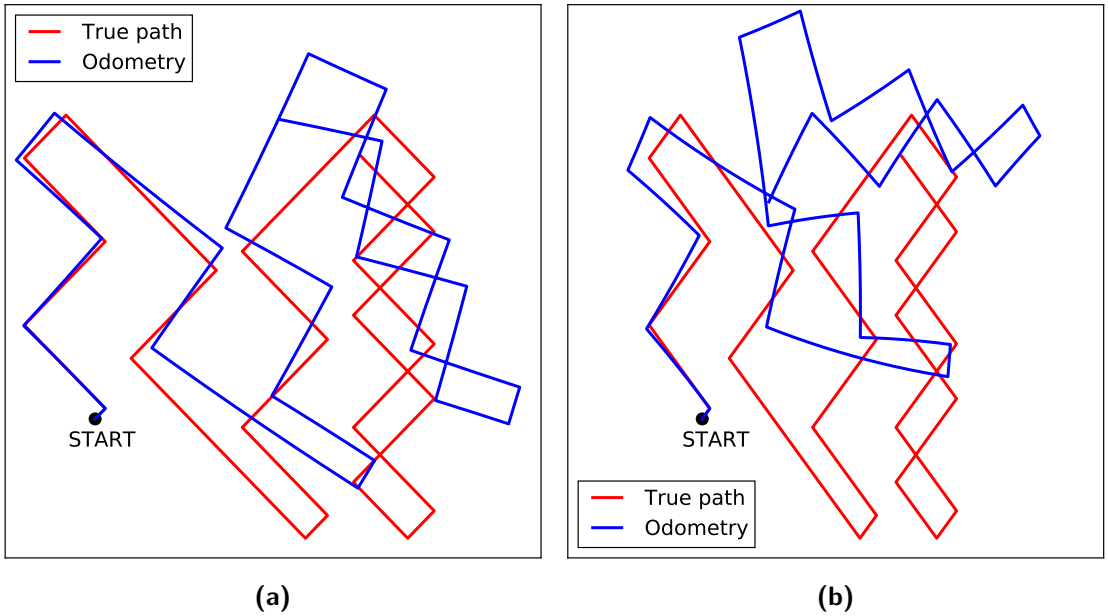


Figure 3-8: Odometry information for the two possible cases of error- translational and rotational. The left figure has a higher translation error while the right figure has a higher rotational error.

noise information where Figure 3-8(a) shows a higher translation drift than rotational drift while Figure 3-8(b) shows a higher rotational drift than translational drift.

The noise in the odometry and collision data have to filtered out against a model using a suitable estimator. The previous sections in this Chapter laid out a suitable algorithm for the impact-based SLAM where the odometry and collision data are filtered to obtain a suitable map of the environment along with the current robot pose (or the entire trajectory). A particle filter is initialized with $M = 10$ particles with the same initial state. The initial pose is trivially taken to be as the origin unless an exact pose is needed to plot the map on a figure window. The initial uncertainty is taken to be very small (zero if possible) to get a more consistent map. The size of the particle set M may vary depending upon the noise covariance of the measurement, including the odometry.

In the prediction, a new set of particles are drawn from the (modified) proposal distribution. In the case of a modified proposal distribution, the particles are drawn from the modified proposal only when a measurement is made available and for rest of the time, the old proposal distribution is used.

As discussed earlier, suitable environment representations for impact-based SLAM are point landmarks and histogram landmarks.

Remark 3.5. *The most suitable environment representation for the SLAM problem depends upon the sensor used and the operating environment. For example, in rectilinear environments, a line representation in polar form $\{\rho, \theta\}$ is most suitable for a robot having a range and bearing sensor.*

The impact-based SLAM uses the histogram landmarks for representing the environment. The landmark states are augmented into each particle while histograms (if needed) are defined

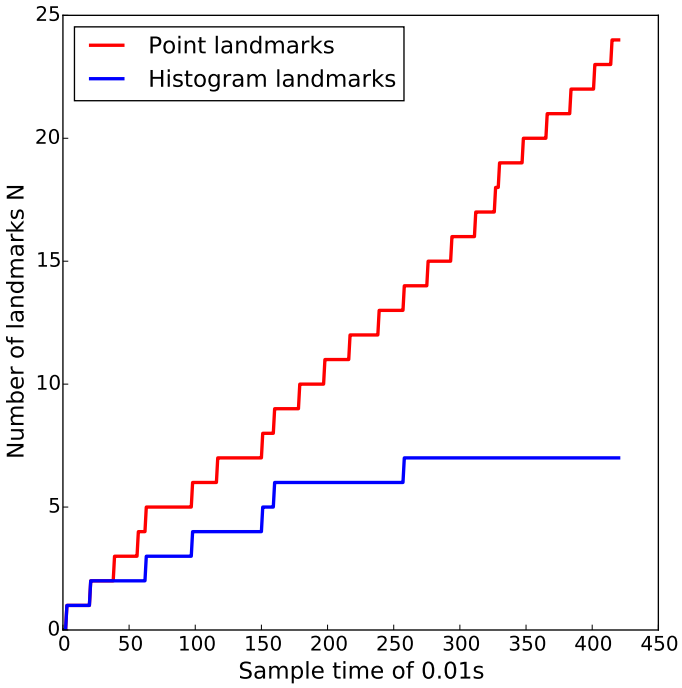


Figure 3-9: Landmark growth for the sample world

as auxiliary variables. Both the landmarks are propagated using EKF while the histogram distribution is propagated using a discrete Bayes filter or a simple histogram update. Nearest Neighbour or χ^2 validation gating is used for associating an incoming measurement to these landmarks. In the case of point landmarks, it becomes difficult to associate point landmarks to new measurements since it is rare for collisions to occur at the same point. Even though the collisions might correspond to the same wall, it is quite difficult to do a loop closure (reobservation of past landmarks). Hence, a new point landmark gets created for every new measurement and there is not a single EKF update. The state space of the SLAM, as a result, grows unbounded in time with a diverging solution in the map (as well as SLAM). This situation does not occur in the case of histogram maps (as discussed in Section 2-4-3) since each wall is regarded as a landmark.

Remark 3.6. *Data association was achieved successfully in a histogram landmark case through a single axis-orientation association. The axis along the wall (shortly called 'wall axis') does not contribute any useful information for data association since the entire wall is regarded as a single landmark while the axis orthogonal to the wall axis is useful for position association. This approach is suitable only for rectilinear environments.*

Figure 3-9 illustrates the growth of landmarks in the SLAM problem for the sample rectilinear world with seven walls. The number of landmarks are bounded in the histogram case due to the finite number of walls in the sample world whereas new landmarks are created for every collision in the point landmark case.

The point landmarks case has a diverging SLAM solution since the robot pose error grows with time. The error covariances of the landmarks remain the same since there are no updates. The histogram map rather has a converging solution to the SLAM problem with each covariance

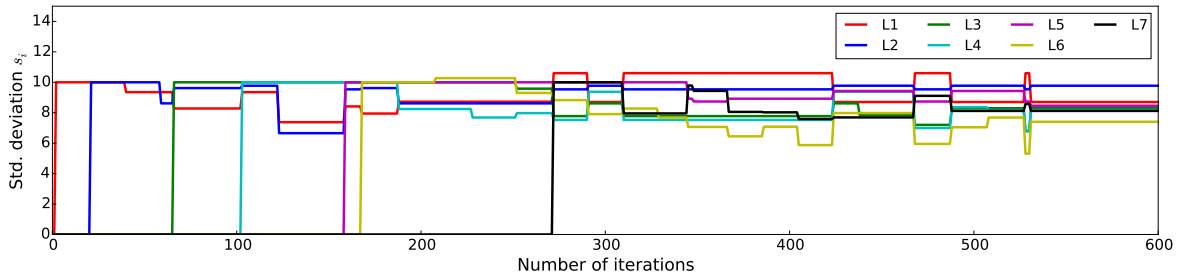


Figure 3-10: Convergence in landmark uncertainty, and as a result, the convergence of the map. A time history of standard deviations for a set of landmark locations are given where each landmark is initialized with a constant conservative estimate.

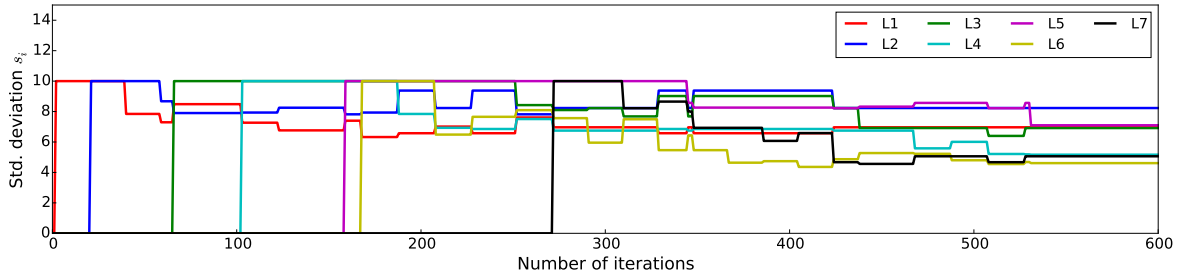


Figure 3-11: Convergence in landmark uncertainty on incorporation of additional EKF-update using rectilinear geometric constraints. Each landmark is initialized with a conservative covariance estimate.

of initialized landmark reducing with time, and is bounded since the covariance cannot get lower than the initial covariance of the robot pose [2]. The convergence in the landmark estimates for histogram maps, and the map as a whole, is illustrated in Figure 3-10. As the SLAM system is observable and correlated, the SLAM solution converges to a true estimate as well.

The incorporation of indoor geometric constraints such as orthogonality, equal corridor width, etc., can result in a consistent SLAM estimation as showed in Section 2-5-2. The constraints are incorporated into the EKF-update step as low (close to zero) uncertainty measurements [16]. Addition of this geometric update results in a more consistent estimate of landmarks and this update is carried out for every new landmark creation (for a non-zero landmark set) and loop closure. The constraints are established only between neighbouring landmarks. The convergence of the landmark estimates are shown in Figure 3-11.

Additional data processing steps are added to output a good consistent map. The histogram data is propagated using a discrete Bayes filter and the data is compared at every update step to a threshold for a consistent data recording since the histogram is propagated as a probability distribution. The old measurements get discounted slowly due to normalization of the distribution which makes the histogram lose past information. As a result, a simple data processing step is used. Figures 3-12, 3-13 and 3-14 illustrate the propagation of histogram distribution with additional data processing steps such as data thresholding and clubbing of collision information in a histogram (with minimum door opening assumption). Figure 3-15 shows the output histogram map of the environment. Note that figure window constraints

(map orientation and position) are incorporated for illustration purposes. For the given environment as well as the real world environment to be used in future, the discretizaton level used is $1/l$ where l is the boundary length. The length can also be set arbitrarily and pruned later.

Regarding the issue of consistency, there is not much reduction in the particle diversity since the impact-based SLAM does not have dense measurements. Hence, the information provided through the sparse measurements, along with frequent loop closures and indoor constraints provide a good estimate of the particles through the modified proposal. As a result, resampling steps were carried out less frequently with the help of adaptive resampling. For situations involving large indoor environments, the frequency of resampling might increase due to the accumulated noise over the measurements which can lead to inconsistency.

A long term inconsistency through resampling can be avoided using multiple robots exploring the environment. The multiple robot scenario can occur in two ways, either robots collect measurements for a period of time and commonly (centralized processing) implement SLAM or the individual robots do a separate SLAM and merge the local maps at a later time. The later approach will be adopted in this thesis.

The maps collected by the individual robots are merged using a simple map comparison with consideration of additional assumptions. The assumptions considered are environment size and approximate starting points for the robot. The multiple robot reduces the exploration time and at the same time improving the local consistency of the solution.

In the simulation, the robots start at the corners of the maze which helps in restricting the possibilities of map comparisons. The robots start to do impact-based SLAM separately and no communication is present between the robots. The robots communicate to a central processing unit where the map merging occurs. The individual maps are merged together once a similarity between the individual maps is found. The merged map generated using two robots are shown in Figure 3-16.

Remark 3.7. *The multiple robot SLAM discussed in this thesis is not a distributed SLAM problem since the merged map does not influence the particle poses of each robot.*

The similarity between the individual maps are found through finite comparisons of histograms. The finite nature of comparisons is attributed due to geometric considerations in the environment that were used for pruning possibilities in the data association along with the additional assumptions.

Summary In this chapter, a detailed algorithm of particle filter-SLAM is provided with the key steps detailed with algorithms and figures. A flowchart is illustrated for clarity of the algorithm. Examples are provided to get an understanding of the landmark initialization and updates for a histogram map. Simulation results are analyzed and issues such as convergence and consistency are addressed.

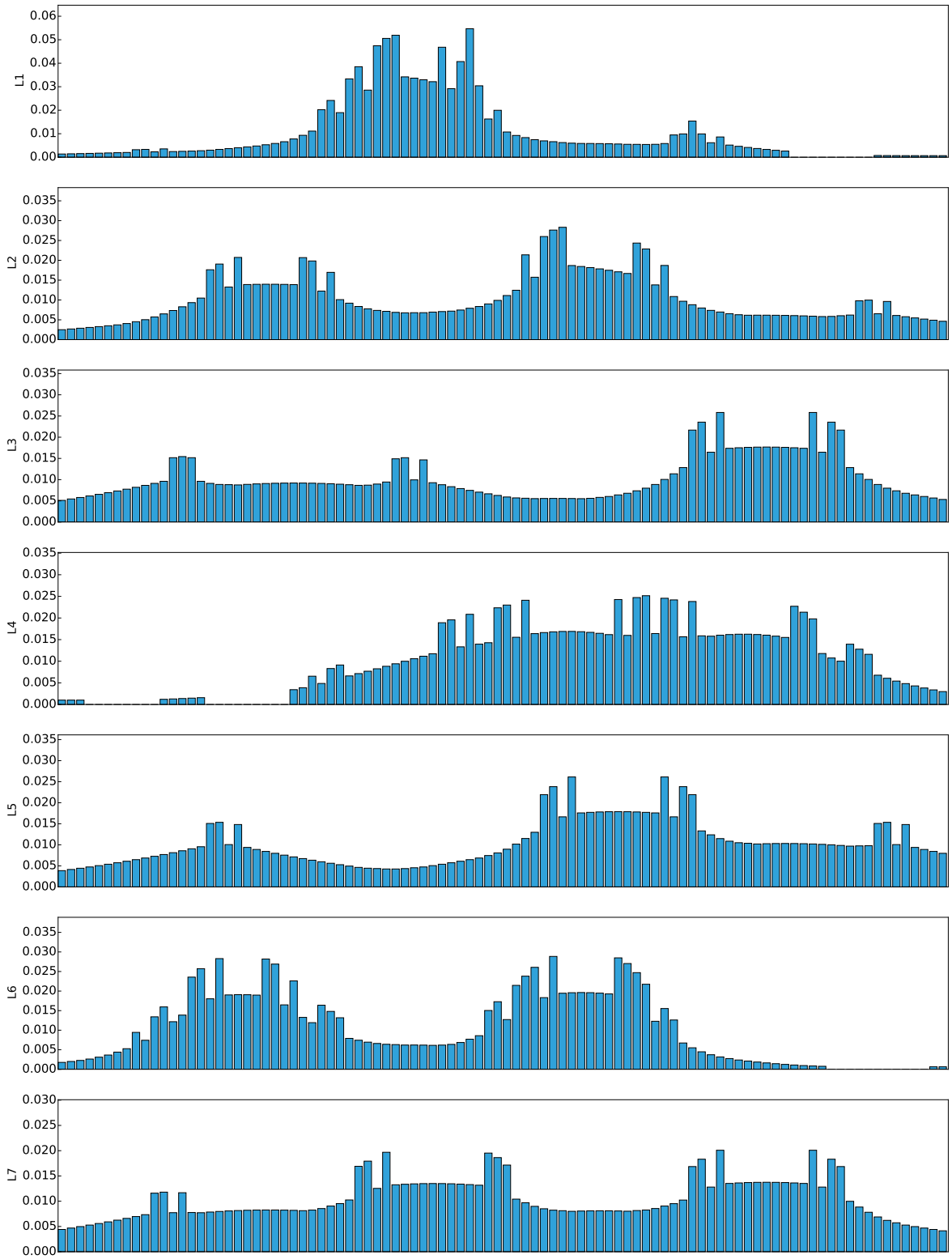


Figure 3-12: Histogram distribution encoding collision information

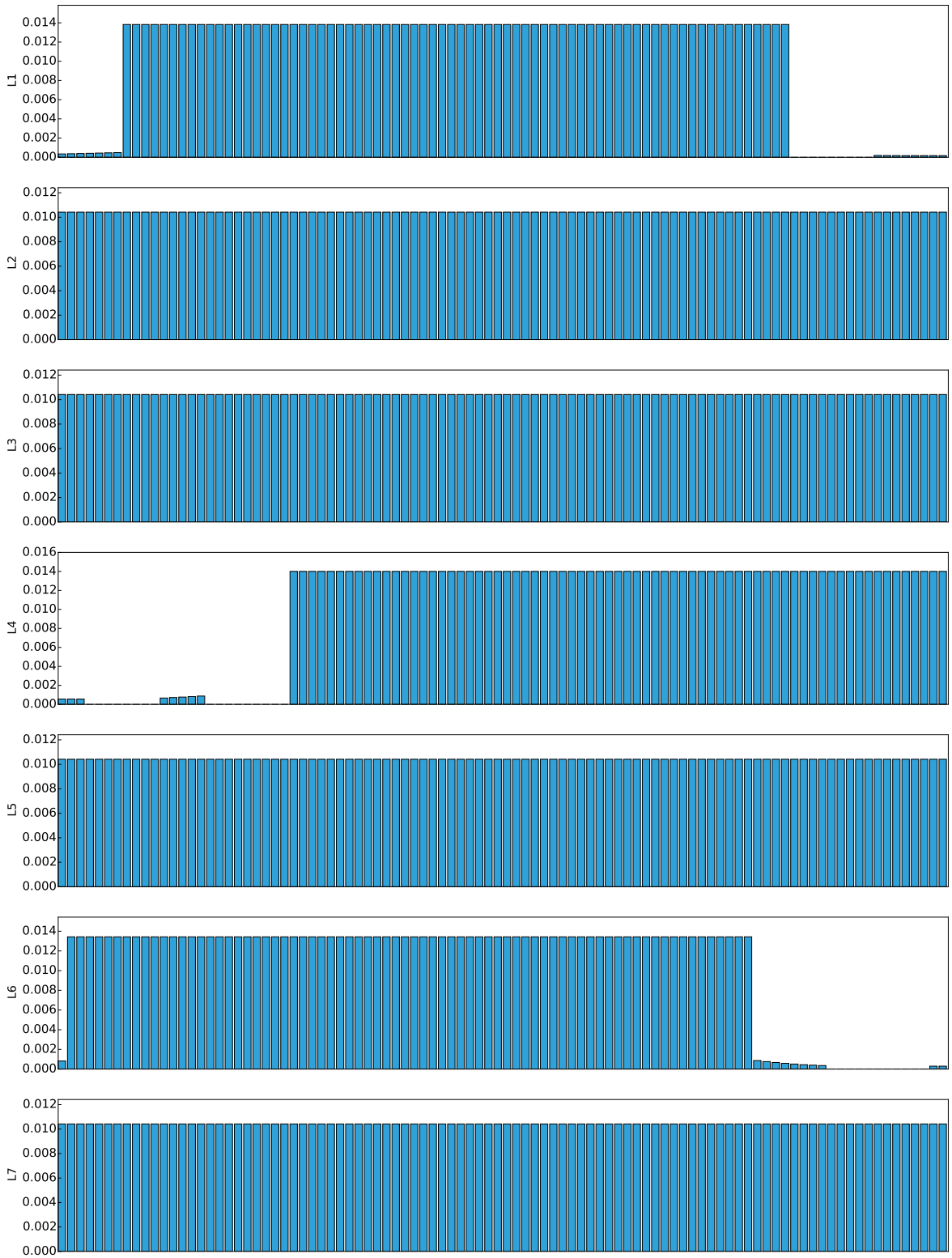


Figure 3-13: Processing of the resulting histograms using data threshold

Figure 3-14: Clubbing of collision information in the processed histogram

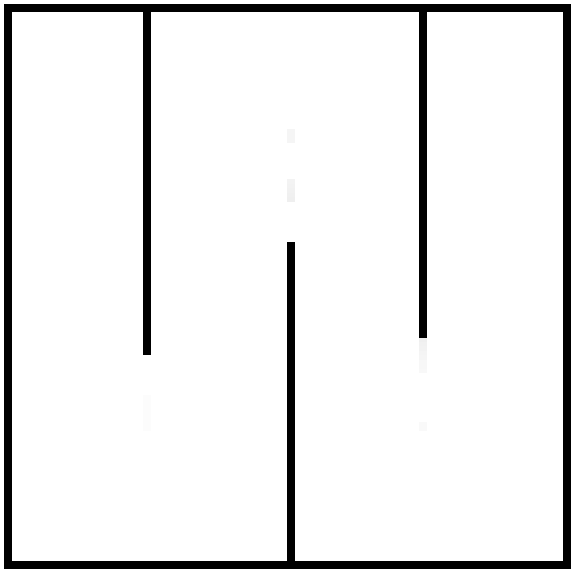


Figure 3-15: Histogram map of the sample world for impact-based SLAM

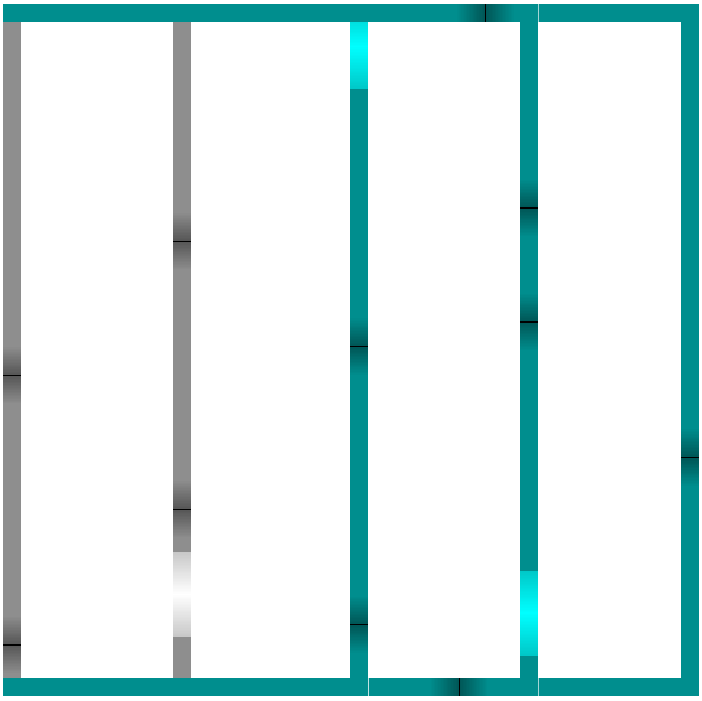


Figure 3-16: Merged map of the sample environment using two robots. Map of robot 1 is shown in gray scale while map of robot 2 is shown in cyan scale.

Chapter 4

Practical implementation

In this chapter, implementation aspects of impact-based SLAM using Sphero robots is presented. Section 4-1 discusses sensing modalities and characteristics of Sphero robotic ball specific to impact-based SLAM. The practical results of SLAM using a single robot are provided in Section 4-2, followed by map-merging using multiple robots in Section 4-3.

4-1 Sphero robot

Sphero is a robotic ball developed by Orobotix Inc. and is essentially a segway robot enclosed in a hollow spherical shell (See Figure 4-1). As the segway moves using its rubber tires, the friction between the shell and tires make the robot move. The unicycle-like robot can spin around the same position and can roll in any commanded direction.

Sphero has been released into the commercial market as a toy robot which can be controlled using a smart phone and an unofficial Software Development Kit (SDK) such as ROS [27] is available for doing project work. A detailed background and technical specifications of the robotic ball can be found in [28].

Sphero is equipped with an onboard Inertial Measurement Unit (IMU) comprising of 3-axis accelerometer and 3-axis gyroscope. This sensor, called a proprioceptive sensor, provides the

Figure 4-1: Sphero 2.0 Robotic ball

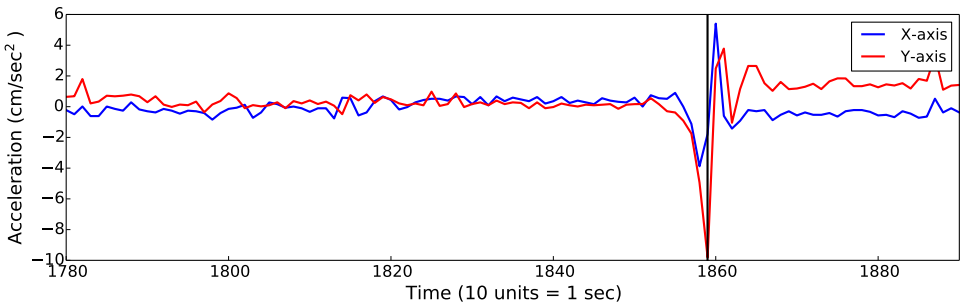


Figure 4-2: Effect of impact on X and Y axis accelerometer data. The ball in this case is forced to be at rest. The collision is detected at the black vertical line.

sole sensing information for solving impact-based SLAM. As said before, achieving SLAM using minimal sensing information is one of the motivations for this thesis.

To solve SLAM, information about the robot's pose as well as from the landmarks in a given environment is necessary. The pose information from the robot is acquired from IMU through dead-reckoning. However, this information is subject to drift due to various error sources such as

- Limited resolution during integration (time increments, measurement resolution)
- Internal wheel slip with shell

These error sources have to be filtered out using a filter provided the error accumulated in the state estimates are independent of each other.

The information from the landmarks in a given environment is obtained using the same IMU sensor. Landmarks are observed through collision detection where Sphero collides with the walls of the environment randomly. Example 4.1 shows the collision detection technique used for impact-based SLAM.

Example 4.1. The collision detection in Sphero is obtained by monitoring the changes in accelerometer data. Figures 4-2 and 4-3 show accelerometer-time plots from Sphero when a collision detection occurs. At the instant of collision, the accelerometer data dips due to deceleration since the robot comes to rest momentarily. The impact force applied by the robot on the wall is given by the equation,

$$F = m \cdot a$$

where m is mass of the robot and a is the acceleration.

The reaction force experienced by the robot from the wall is ideally same as the impact force and the rate of change of this reaction force (equal to 'jerk') is used for computing the wall orientation.

As the mass m is a constant, wall orientation can also be computed equivalently from the rate of change in acceleration along the X axis and Y axis. For this example, the rate of change in acceleration is computed from the accelerometer plot (Figure 4-3) and the ratio is used to infer the new heading (refer Equation 3-9).

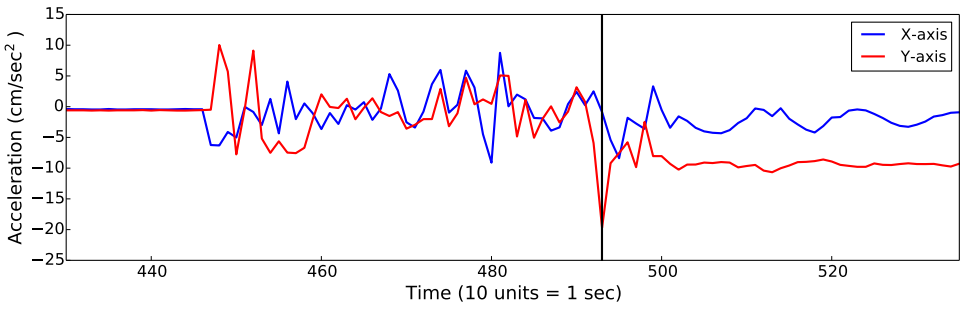


Figure 4-3: Effect of collision on X and Y axis accelerometer data. The ball with a non-zero initial velocity collides against a wall. The collision is detected at the black vertical line.

The initial robot orientation for the moving robot example in Figure 4-3 is 0 radians. The new heading is found from the left/right handed collision model and is close to π radians for $c = 0$.

4-2 Single robot SLAM

The particle filter-SLAM algorithm, developed in Chapter 3, uses the robot's odometry and collision measurements. The pseudo-code for the impact-based SLAM algorithm is provided in Appendix C. The real world test was carried out with a particle size $M = 30$ and an increased accuracy is observed from $M = 5$ to $M = 25$. The experiment and computations were carried out on a 1.6GHz Xeon workstation and the measurements are provided at a frequency of 16Hz.

The control action applied to the robot is in a random fashion with the next action only depending upon the inference from the current wall collision. The control inputs or actions are robot speed and heading angle, out of which the former is not controlled and is set to a constant value depending upon the operating conditions such as size of the environment. The latter one, heading angle of the robot, must be modified after every collision and as said, the angle is chosen randomly from a given set or it can be assigned using Equations 3-8 and 3-9 with a pre-determined value of constant c.

From the motion models described through a FSM (Figure 3-2), the constant c has to be defined for impact-based collisions. A simple strategy is to go for a reflective collision (angle of incidence is same as angle of reflection) in ball movements ($c = 2$) which was followed in the simulations. This approach was found to be very poor in practical scenarios due to huge orientation error accumulating after every collision. The main reason for this orientation error is the slip between the segway-like robot and the hollow shell. The robot heading is found to differ from the actual orientation (yaw axis) by atleast 2π for 3 to 5 consecutive collisions. Calibration of this error is difficult to be performed online since it involves destabilizing the robot which is time consuming and prone to position errors[1]. Hence, the robot heading is restricted to a set of values which has the inner slip-less (ideally) collision and this approach is found to have a lower orientation error, suitable for practice. The set of possible headings

[1]Destabilization involves making the robot stationary and during this phase, the robot exhibits an unusual motion that gets accumulated into the odometry.

Figure 4-4: Real world environment for impact-based SLAM

is unit normal directions of wall hyperplanes where the initial orientation is computed from the first collision. However with head-on collisions, the robot has a higher orientation error from drift in IMU when the robot turns around by π radians.

With random control inputs from the set, impact-based SLAM is implemented for an indoor environment of dimensions 280cm $\times$ 280cm as shown in Figure 4-4. The maximum likelihood solution is obtained by choosing particle with highest importance weight in the particle set. The resulting output map of SLAM solution is shown in Figure 4-5.

In the output map, the walls are orthogonal to each other and an initial local loop (upto neighbouring landmarks) closure is performed correctly for the robot starting from bottom-right corner of the maze. The initial local loop closures can be seen from the neighbouring landmarks that are accurately estimated (black coloured landmarks). The orthogonal constraints are added as measurements with a low covariance (close to zero) at every correction step through an additional EKF update [29], [16]. This constrained estimation projects the prior estimate onto the constraint space which is same as projection operation in least squares framework. However, the approach yields inconsistent estimates here (as seen from the spurious landmark growth) due to lack of proper modelling of correlations between all the landmarks and robot. Inconsistency can also arise through linearization errors which is discussed in detail in [29] and [24].

In this thesis, landmarks are assumed to be independent (no cross-correlations) through Rao-Blackwellization which may not hold in structured environments. This is because the incoming measurements are correlated through the geometric structure of the environment apart from the robot pose error. The lack of maintaining these cross-correlations accurately is a key issue for inconsistency since an EKF update uses the correlations (covariance matrix) for updating the state estimate. As the correlations are modelled inaccurately, addition of constraint

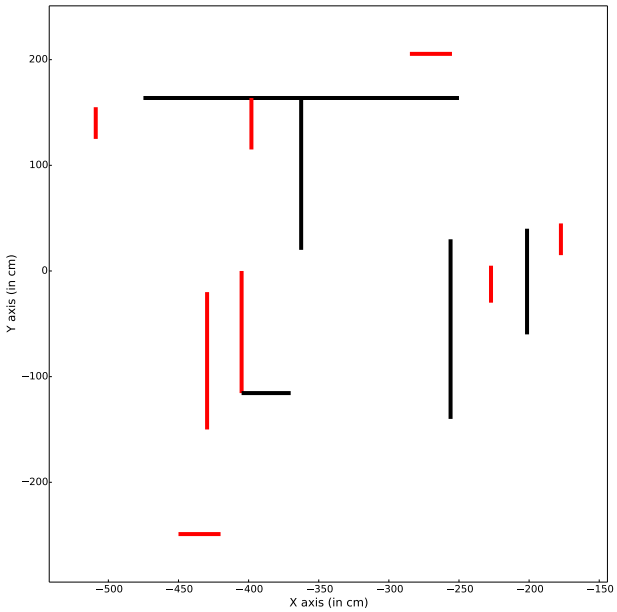


Figure 4-5: Output map of SLAM solution with constraint incorporation as measurement with low covariance. The red coloured landmarks are spurious and are not updated. The histogram distribution is processed finally and pruned to the nearest neighbouring landmarks.

measurements with very low covariance can result in a large reduction in covariance which results in an inconsistent solution. The large reduction in covariance for landmark updates can be seen in Figure 4-6. Hence with few measurement (including constraint) updates, the covariance spuriously drops down or equivalently, the estimate becomes more certain. Hence, the later updates (no measurement updates after 600 iterations) does not contribute to any new information as seen from the lack of larger loop closures. The effect of correlations on the state estimate for the SLAM problem is asserted in Remark 4.1.

Remark 4.1. *For a small magnitude in cross-correlation between state estimates in SLAM, incoming measurements are treated with a higher novelty which updates covariance information spuriously. A large magnitude of cross-correlation results in a rigid covariance structure which makes an estimate more susceptible to error of other correlated estimates.*

The loss of consistency can be seen from the Figure 4-5 through lack of a large loop closure. This can also be seen through creation of spurious landmarks (a spurious local map) after traversal of a large loop. The spurious creation is due to formation of a new local map of the same region which was mapped long time back.

The covariance of the state estimates with a low covariance constraint update is shown to converge to a bounded value as shown in Figure 4-6. It is observed that there are not much landmark updates since new spurious landmarks are created due to lack of a consistent estimation. As covariance Σ drops down, past landmarks become more certain and data association gets affected. This is verified from Equation 2-29 where magnitude of a likelihood to any measurement reduces as covariance drops down. This also holds for other state-of-the-art data association techniques based on associating measurements using covariance information. As data association gets poor, new landmarks are created for measurements with lower noise.

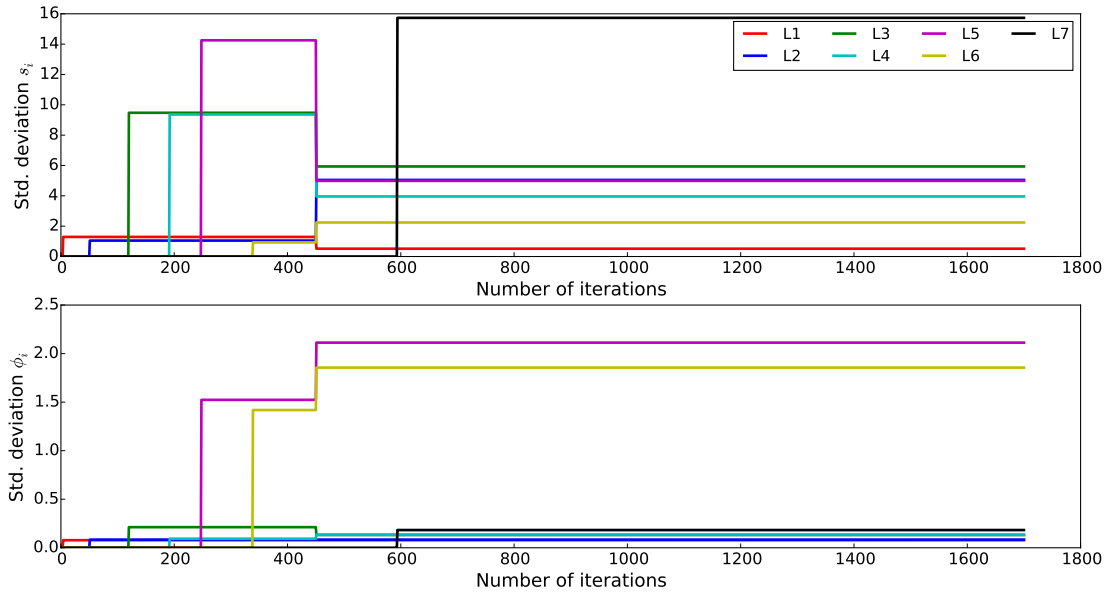


Figure 4-6: Convergence plot of the SLAM solution with constraint updates of low (close to zero) covariance. The standard deviation (1σ) plots for the (along unit normal axis) position s_i and orientation ϕ_i of landmarks are computed from the ML particle.

An effective approach for improving the consistency is through modelling the correlations using a single joint covariance estimate for all the landmarks (a single large covariance matrix for every particle in particle filter-SLAM) in the map and propagating it over the incoming measurements. The joint covariance maintains correlations due to geometric structure as well as the correlations introduced through robot pose error. However, the use of a joint covariance matrix involves higher computational complexity and storage requirements ($\mathcal{O}(M \cdot N^2)$ where N is the number of landmarks in the map). Though conditional independence in landmark states through Rao-Blackwellization is lost due to the geometric correlations, higher consistency can be achieved at the expense of computational efficiency.

An alternative approach for ensuring consistency and at the same time exploiting the Rao-Blackwellized property is to update the constraints in a a conservative fashion using Covariance Intersection (CI) where the landmark pose estimates are updated with the constraint measurements with a non-zero covariance. This approach does not take the cross-correlations into account but yields a consistent estimate. However with this approach, it is difficult to compute the constraint measurement covariance at every time step accounting for the loss of correlations. In the practical implementation, the covariance magnitude is chosen heuristically based on the current magnitude of orientation error.

The convergence to a more accurate solution is achieved using covariance intersection. The number of spurious landmarks are reduced since more loop closures (with a longer loop) are performed as seen from the updates. For impact-based SLAM with constraint updates using covariance intersection, the convergence plots are shown in Figures 4-7 and 4-8. The convergence to a consistent solution shows the success of a SLAM solution.

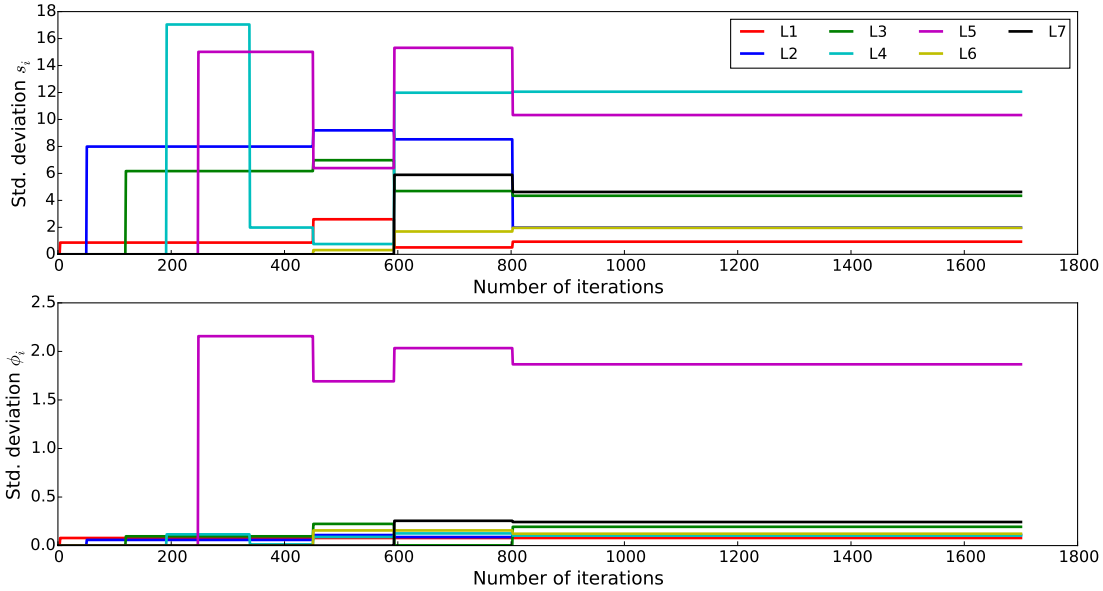


Figure 4-7: Convergence in the map solution for a conservative constraint update of covariance $R_{CI} = 0.1 \cdot \Sigma_\theta$. The standard deviation (1σ) plots for the position s_i and orientation ϕ_i of landmarks are computed from the ML particle.

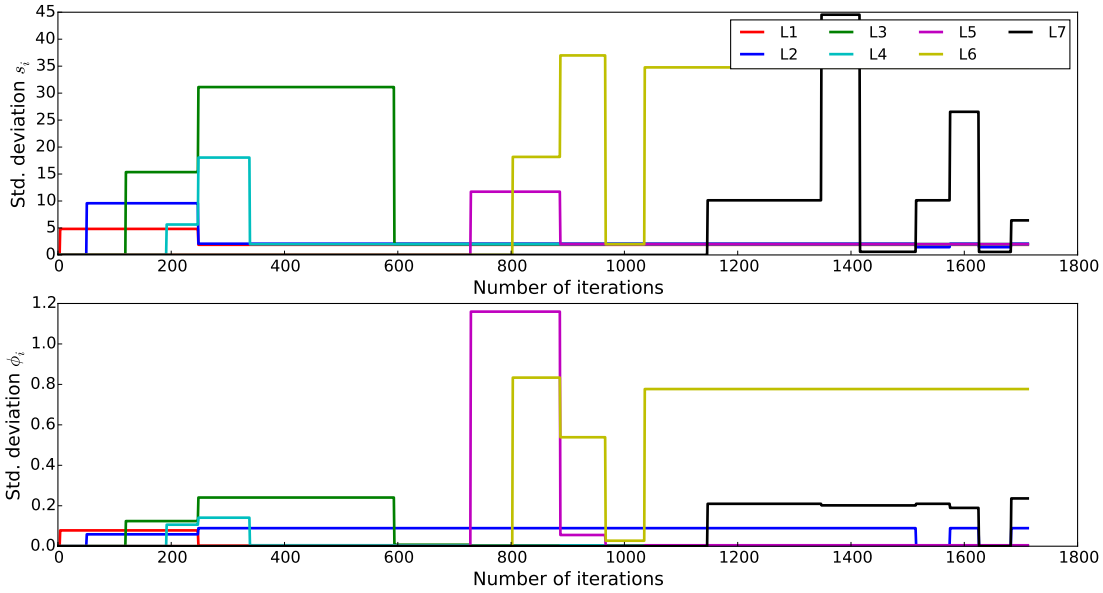


Figure 4-8: Convergence in the map solution for a conservative constraint update of covariance $R_{CI} = 0.5 \cdot \Sigma_\theta$. The standard deviation (1σ) plots for the position s_i and orientation ϕ_i of landmarks are computed from the ML particle.

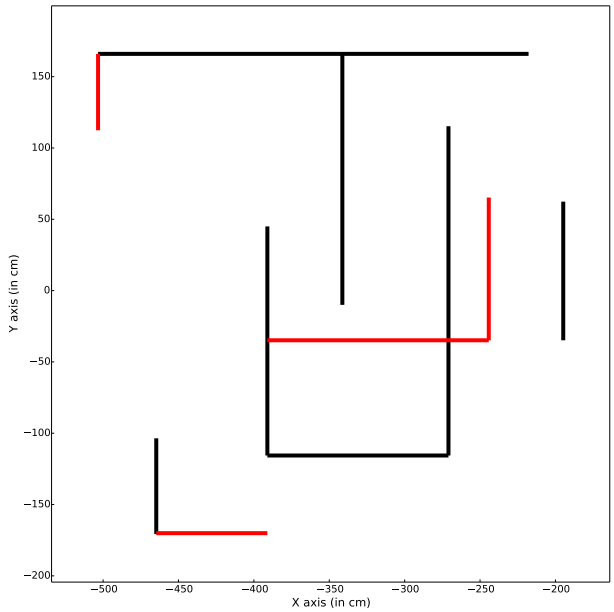


Figure 4-9: Output map from a conservative SLAM solution using a single robot corresponding to constraint measurement covariance of $R_{CI} = 0.5 \cdot \Sigma_\theta$. The red coloured landmarks are spurious landmarks initialized due to high motion error or false positives in collision. The histogram distribution is processed finally and pruned to the nearest neighbouring landmarks.

Remark 4.2. *The standard deviation (or covariance) increase is due to jumps between particles for plotting the most likelihood solution and the jumps are higher for a higher orientation error.*

The resulting output map of SLAM using the conservative constraint update is shown in Figure 4-9. The conservative approach using CI results in a solution with a higher consistency as seen from the reduced growth of spurious landmarks. It can also be seen from Figure 4-8 that landmarks are updated more frequently which suggests that better loop closures are achieved. Problems arising from bigger loop closure can only addressed with much better data association techniques such as Joint Compatibility Branch and Bound (JCBB) where multiple landmarks are considered simultaneously for association based on incoming measurements. The problem with impact-based SLAM is the lack of dense measurements from the environment and hence for considering Joint Compatibility, past subset of measurements have to be incorporated at every update step.

An additional conservative update is carried out to account for the partiality in uncertainty update in a rectilinear environment. In a corridor environment, the covariance gets selectively updated along the direction of unit normal to the wall (or along the corridor width) while the covariance along the wall (or corridor path) increases due to process noise. This can be seen in Figure 2-9 where error along the corridor gets accumulated until orthogonal information is available through a loop closure of a wall orthogonal to the corridor. This issue arises from the geometric structure of the environment and may be severe in environments with long corridors. As the landmarks are made independent through Rao-Blackwellization, a CI update is carried out between neighbouring orthogonal landmarks once a loop closure is achieved along the orthogonal direction. This update also helps in an accurate histogram

update since the histogram update is influenced by the covariance of the landmark position (coordinate axis along the wall).

The histograms of landmarks are pruned to the nearest neighbours and the histograms along the perimeter are pruned based on environment size assumption. The discretization or rasterization level of histograms depends upon minimum wall opening (such as doors) assumption to speed up robot exploration. A key advantage about using histogram maps is collinear consistency of the output map which is not present in other map representations such as point landmarks and occupancy maps.

Other issues related to the practical impact-based SLAM are false positives generated by Sphero and corner collisions. The false positive detections of Sphero collision is reduced by increasing the impact threshold (rate of acceleration) on the accelerometer axes. In addition, a more effective way to reduce false positives is through a postponed association where landmarks are included in the map only if they are detected more than certain times. Few ambiguous collisions can occur when the robot hits a corner and such measurements are not considered in the estimation.

4-3 Multiple robot SLAM

The impact-based SLAM can be extended to multiple robots as a map-merging problem where individual maps are collected and merged at a point when a match is found. The histograms are matched using simple map comparisons with the assumption of environment size and approximate initial starting points of the robots. In addition to the map-merging, the assumptions are also used to prune length of histogram much similar to the single robot scenario.

With use of multiple robots for exploration, the merged-map is more consistent than single robot-SLAM since local maps are subjected to lower linearization errors. The effect of inconsistency through lack of accurate correlations is less observed in multiple robot scenario since local maps have lower number of updates but for generality, covariance intersection technique for constraint measurements is followed. Usually, multiple robots achieve faster exploration compared to a single robot exploration with a higher consistency. This can be seen in Figure 4-11 through reduction in spurious landmarks

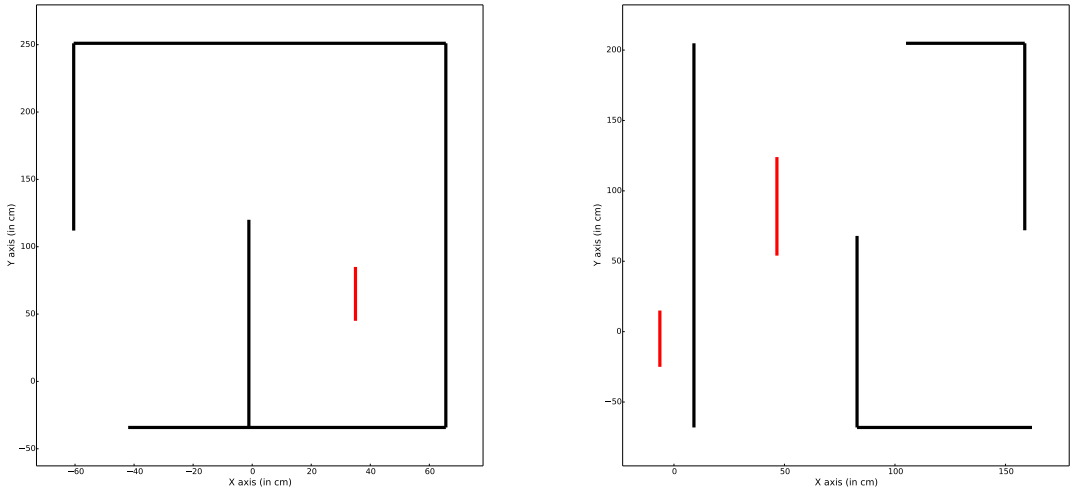
Covariance intersection is used for fusing the estimates of the intersecting (common) landmarks of the local maps and their histogram data is merged. For example, let (x_1, P_1) and (x_2, P_2) be the estimates of an intersecting landmark from the local map of robot 1 and 2 respectively. The merged estimate is found according to the following equations,

$$x_{CI} = P_{CI}\left(\omega \cdot P_1^{-1} x_1 + (1-\omega) \cdot P_2^{-1} x_2\right) \tag{4-1}$$

$$P_{CI} = \left(\omega \cdot P_1^{-1} + (1-\omega) \cdot P_2^{-1}\right)^{-1} \tag{4-2}$$

where ω is obtained by solving the following optimization using Golden Section search technique.

$$\min_{\omega} tr\left(\omega \cdot P_1^{-1} + (1-\omega) \cdot P_2^{-1}\right)^{-1} \tag{4-3}$$



(a) Local map generated by robot 1. The red landmark denotes a spurious landmark

(b) Local map generated by robot 2. The red landmarks denote spurious landmarks

The output maps generated by two robots exploring the same environment are shown in Figures 4-10(a) and 4-10(b). The two maps closely represent the actual environment as shown in Figure 4-11 and must be merged ($\mathcal{O}(1)$ complexity) consistently. The merged map is shown in Figure 4-12 with histograms merged as well. An intersecting landmark pose has to be common among both the maps for fusion. It can be seen that local consistency is quite good from the convergence as well as the reduced number of spurious landmarks. However, the merged map is globally less consistent as seen from the unequal corridor width since the local maps of the robot are not correlated with each other. A possible approach (future work) to improve the global consistency is through continued exchange of particle information (fusion of two point clouds) during exploration between the robots.

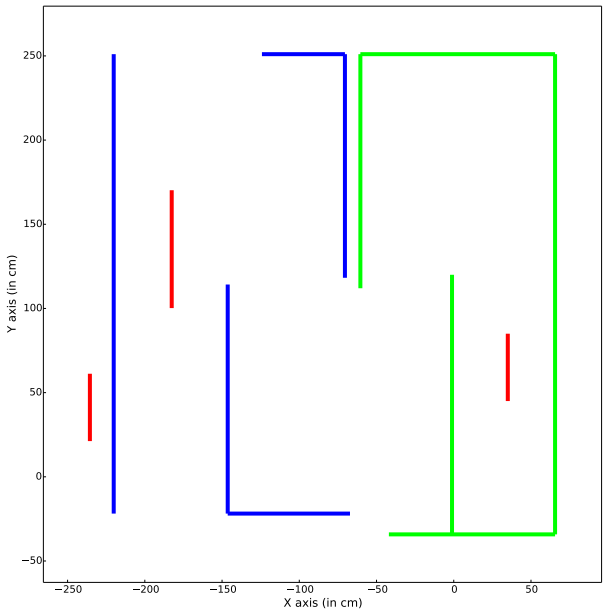


Figure 4-11: Local maps generated by the two robots. Each local map is denoted either by red or green color and the red landmarks correspond to spurious landmarks. Note that top horizontal landmarks of both the maps are pinned together for comparison.

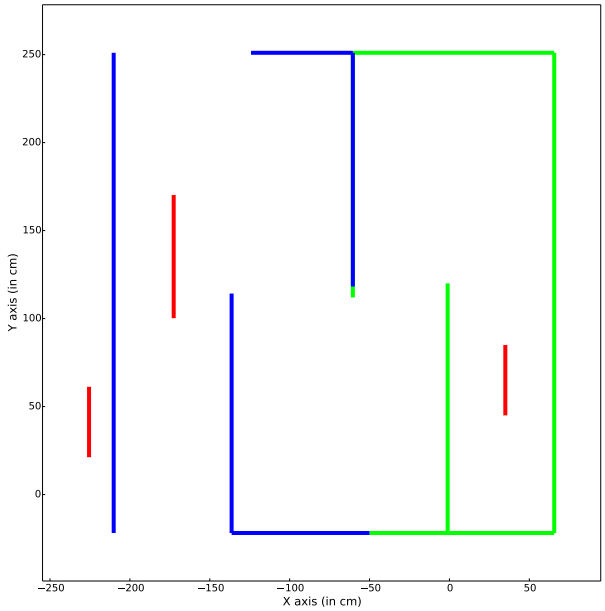


Figure 4-12: Merged maps from the two robots. Each local map is denoted either by red or green color and the red landmarks correspond to spurious landmarks. Note that the pose of vertical intersecting landmark are pinned together.

Chapter 5

Conclusions and Future work

5-1 Summary

In this thesis, an impact-based approach to Simultaneous Localization and Mapping is presented for a rectilinear environment and implemented on a practical setup using Sphero robots. A new environment representation, called histogram map, is adopted to the SLAM problem which is suitable for the sensor used and operating environment. The robot trajectory is estimated using a probabilistic technique called particle filter and landmarks are estimated using low dimensional EKFs. A multiple robot scenario is used to reduce the exploration time to reduce the sparse nature of incoming measurements.

The use of a particle filter is favoured due to the non-Gaussian robot pose distribution. The poor pose distribution was attributed to the nature of impact-based SLAM but has an useful advantage that no other SLAM solution can provide. The use of impact information with the landmarks gives the absolute location information of landmarks which makes the impact-based SLAM problem observable.

For the particle filter, a modified prediction step is adopted since the measurements arrive in a sparse fashion with a higher accuracy than the robot motion. This avoids degeneracy in the particles generated, and thereby improving the consistency of the estimate. The motion of the robot is modelled as a unicycle along with a Finite State Machine to model the effect of collisions. The motion model accounts for Markov assumption for computational efficiency. Similar assumptions hold for the incoming sparse measurements such as independence of a measurement on old states.

The sparse measurements comprise of the collision information which is interpreted through the accelerometer data. The measurement is associated with an existing landmark on a per-particle basis using a χ^2 acceptance test or Nearest Neighbour. This approach is found to be robust since associations are carried out per-particle. If an association is satisfied for an existing landmark, the corresponding landmark is updated using EKF, otherwise a new landmark is created. For histogram maps, a new histogram is associated for every new landmark. In addition to the collision information, the rectilinear world constraints such as

Table 5-1: Comparison Results

Number of robots	Convergence	Consistency	Exploration time	Computational effort
1	Yes	Moderate	107.18 sec.	$\mathcal{O}(MN)$
2	Yes	High	67 sec.	$\mathcal{O}(MN)$

orthogonality, equal corridor width provide good information on the resulting map of the environment. These constraints are incorporated as a geometric update through constrained estimation in EKF. In simple words, the constraints are incorporated for the new landmark as a zero-uncertainty measurement in EKF. Negative information is also incorporated into the algorithm for improving the accuracy of the output map. An adaptive resampling is used to remove erroneous hypotheses in the particle set, thereby improving the accuracy of the estimates and also ensuring particle diversity at the same time.

The resulting algorithm in the worst case scales linearly with the number of landmarks. The spurious growth of point landmarks is eliminated using a suitable map representation for impact-based SLAM. SLAM using histogram maps is found to be very efficient in terms of number of landmarks and influencing the efficiency of data association and consistency of the solution. Additional geometric assumptions such as pruning of associations, geometric wall locations, etc. are exploited for improving the efficiency of the algorithm.

The developed SLAM algorithm is implemented on a practical setup using Sphero robots. An offline version of SLAM was accomplished under manual calibration since the Sphero suffered a severe internal slip after every collision. An online version of SLAM was accomplished only through head-on collisions. Results conclude that impact-based SLAM with minimal sensing information can be accomplished under usual rectilinear world assumptions.

A tabled conclusion of the accomplished SLAM algorithm in the single and multiple robot scenario is shown in Table 5-1.

5-2 Future work

In this section, recommendations for future work are listed out. Regarding the impact-based SLAM algorithm, following can lay some foundations for future work.

1. A consistent map estimation can be achieved by properly maintaining the structural correlations of the map and the effect of this inconsistency can be severely observed in more structured situations. The need for maintaining a joint covariance matrix can be studied in detail and suitable exploitations in the covariance structure can be found to improve the computational efficiency in such a case.

2. The multiple robot SLAM problem can be converted to a distributed SLAM where each system state can influence the other after map merging. An optimal method for map fusion in particle filters is a great area to explore upon.

3. The spatial information of the wall presence is indicated through a histogram in this thesis. The histogram representations can be a key ingredient for the planning aspects since the areas to be explored can be defined using the histogram information.

About the practical implementations, an interesting scope of work can be done. The SLAM code can be optimized to incorporate into the robot's memory such that the computations can be done onboard. Few works in the literature implemented an optimized SLAM algorithm which can be implemented onboard [30], [31].

Appendix A

Rao-Blackwellized factorization

Particle filters have a great advantage of representing any arbitrary distribution through its non-parametric representation of distribution and it is simple and intuitive. A downside in use of particle filters is the exponential scaling of particles with the dimensions of state space. Hence, the use of particle filters have been restricted to low dimensional state estimation and tracking.

However, few structural properties of the system can be exploited to improve the computational efficiency of the filter. In the SLAM scenario, the conditional independence property, as stated in Section 3-1, can be exploited and this factorization is called as Rao-Blackwellized factorization. The nature of this factorization is exact and landmark independence assumption (as in Section 2-5-2) is taken into account in this following derivation.

From product rule, SLAM posterior can be rewritten as,

$$p(x_{1:t}, m|z_{1:t}, u_{1:t}, c_{1:t}) = p(x_{1:t}|z_{1:t}, u_{1:t}, c_{1:t}) \cdot p(m|x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \tag{A-1}$$

To obtain the factored version of SLAM posterior, it is sufficient to show the following-

$$p(m|x_{1:t}, z_{1:t}, c_{1:t}) = \prod_{n=1}^{N} p(m_n|x_{1:t}, z_{1:t}, c_{1:t}), \tag{A-2}$$

and derived through induction. Firstly, the Bayes rule is applied to a feature observation given the robot trajectory $x_{1:t}$, controls $u_{1:t}$ and observations $z_{1:t}$.

$$p(m_i|x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) = \frac{p(z_t|m_n, x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t})}{p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t})} p(m_i|x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t}) \tag{A-3}$$

As in the last term in the product, the observed feature does not depend upon the current pose and control unless with the current observation. Applying Markov assumption to the above product,

$$p(m_i|x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) = \frac{p(z_t|m_n, x_t, c_t)}{p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t})} p(m_i|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1}) \tag{A-4}$$

Assuming the following induction hypothesis, which trivially holds true for $n = 1$,

$$p(m|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1}) = \prod_{n=1}^{N} p(m_n|x_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \tag{A-5}$$

The induction will hold true for $n > 1$, by taking a general posterior and applying the same set of operations,

$$p(m|x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) = \frac{p(z_t|m_n, x_t, c_t)}{p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t})} p(m|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1})$$

Using the induction principle,

$$= \frac{p(z_t|m_n, x_t, c_t)}{p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}, c_{1:t})} \prod_{n=1}^{N} p(m_n|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1})$$

Using Equation A-4,

$$= p(m_i|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1}) \prod_{n \neq i}^{N} p(m_n|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, c_{1:t-1})$$

$$= \prod_{n=1}^{N} p(m_n|x_{1:t}, z_{1:t}, c_{1:t})$$

Hence, the exact factorization is proved.

Appendix B

Modified proposal distribution

A modified proposal distribution was proposed by [12], [32] to overcome the particle degeneracy problem in situations where the robot's motion is more noisier than incoming measurements. In this situation, the particles are drawn from a noisy motion model which distributes the particles over a relatively large space. This scenario occurs frequently in practical situations including the impact-based SLAM using Sphero robots. This is because a systematic error arises in the motion of the robot as it starts to drift, in addition to other motion noise such as slip. This error gets integrated over time and is accounted as noise in the motion model. The measurement error is comparatively lower than the motion noise in the impact-based SLAM, provided the calibration is taken care. More details on calibration is given in Section 4-1.

Deriving the new proposal

The use of measurements in the motion model samples the next robot pose using a more accurate measurement in addition to the noisy motion model. This does not result in degeneracy of the particle set after few iterations as shown in Figure 1-12. The use of modified proposal distribution improvises the prediction step calculation through consideration of measurement by taking into account for the measurement variance propagation.

The modified proposal distribution can be written as,

$$p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) \tag{B-1}$$

The proposal distribution can be expanded using Bayes rule as follows,

$$p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) = \eta \cdot p(z_t|s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \tag{B-2}$$

Taking the Markov assumption on second term of the product which is the old motion model,

$$p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) = \eta \cdot p(z_t|s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(s_t|s_{t-1}^{[k]}, u_t) \tag{B-3}$$

Using the theorem of Total Probability where the measurements are conditioned upon the current observed landmark c_t,

$$= \eta \cdot \int p(z_t|m_{c_t}, s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(m_{c_t}|s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot dm_{c_t} \cdot p(s_t|s_{t-1}^{[k]}, u_t) \tag{B-4}$$

Applying Markov assumption,

$$= \eta \int p(z_t|m_{c_t}, s_t, , c_t) \cdot p(m_{c_t}|s_{t-1}^{[k]}, u_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \cdot dm_{c_t} \cdot p(s_t|s_{t-1}^{[k]}, u_t) \tag{B-5}$$

The first term of the integrand is the old measurement model while the second term is the landmark model. Both the terms are expressed as Gaussian for a closed form solution. The same holds for the motion model. The integrand can be solved using convolution with a linearized measurement model. A first order Taylor expansion of the measurement model (refer Section 3-2-4) is shown below,

$$z_t \approx \hat{z}_t + G_s \left(s_t - \hat{s}_t\right) + G_m \left(m_{c_t} - \mu_{c_t,t-1}^{[k]}\right) \tag{B-6}$$

The convolution can now be obtained for the terms inside the integrand as follows,

$$\mathcal{N} \left(\hat{z}_t + G_s \left(s_t - \hat{s}_t\right), \underbrace{R_t + G_m \Sigma_{c_t,t-1}^{[k]} G_m^T}_{Z_{t,c_t}} \right) \tag{B-7}$$

The proposal distribution is obtained through the product of the above Gaussian and Gaussian model of motion $\mathcal{N}\left(\hat{s}_t, P_t\right)$. The above Gaussian models the propagation of measurements into the modified proposal distribution, as seen in the first term of the addition in the following χ^2 distribution. The second term of the addition refers to the propagation of motion noise. The modified proposal Gaussian can be written as,

$$p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) \propto \exp\left(-\chi^2\right) \tag{B-8}$$

where,

$$\chi^2 = \frac{1}{2}\left[\left(z_t - \hat{z}_t - G_s s_t + G_s \hat{s}_t\right)^T Z_{t,c_t}^{-1} \left(z_t - \hat{z}_t - G_s s_t + G_s \hat{s}_t\right) + \left(s_t - \hat{s}_t\right)^T P_t^{-1} \left(s_t - \hat{s}_t\right)\right] \tag{B-9}$$

The mean and covariance of the new sampling distribution can be obtained by minimizing the χ^2 distribution through first and second derivatives, much similar to the derivation in Section 3-2-4.

$$\Sigma_{s_t,c_t}^{[k]} = \left(G_s^T Z_{t,c_t}^{-1} G_s + P_t^{-1}\right)^{-1} \tag{B-10}$$

$$\mu_{s_t,c_t}^{[k]} = \Sigma_{s_t}^{[k]} G_s^T Z_{t,c_t}^{-1} \left(z_t - \hat{z}_t\right) + \hat{s}_t^{[k]} \tag{B-11}$$

The Gaussian is constructed for each particle in the sample set, and a new sample is drawn and placed in the temporary particle set which approximates the new proposal distribution.

Calculating importance weights

The importance weight [32] can be calculated using Equation 1-15 with the modified proposal distribution can be written as the product,

$$p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \cdot p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) \tag{B-12}$$

The modified proposal distribution is written as a product of old posterior distribution and the new prediction model while the target distribution is the new posterior, encoded along with the measurement likelihood $p(z_t|\cdots)$.

$$\begin{aligned} w_t^{[k]} &= \frac{\text{target distribution}}{\text{proposal distribution}} \\ &= \frac{p(s_t^{[k]}|u_{1:t}, z_{1:t}, c_{1:t})}{p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \cdot p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t})} \end{aligned}$$

Expanding the numerator,

$$\begin{aligned} &= \frac{p(s_t^{[k]}|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t}) \cdot p(s_{t-1}^{[k]}|u_{1:t}, z_{1:t}, c_{1:t})}{p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \cdot p(s_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t}, c_{1:t})} \\ &= \frac{p(s_{t-1}^{[k]}|u_{1:t}, z_{1:t}, c_{1:t})}{p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1})} \end{aligned}$$

Using bayes rule on the numerator,

$$\begin{aligned} &= \eta \frac{p(z_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1})}{p(s_{t-1}^{[k]}|u_{1:t-1}, z_{1:t-1}, c_{1:t-1})} \\ &= \eta\, p(z_t|s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \end{aligned}$$

Applying Theorem of Total Probability twice, first on the robot pose and second on the associated landmark,

$$\begin{aligned} &= \eta \int p(z_t|s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) \cdot p(s_t|s_{t-1}^{[k]}, u_t) ds_t \\ &= \eta \int \int p(z_t|m_{c_t}, s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) p(m_{c_t}|s_t, s_{t-1}^{[k]}, u_{1:t}, z_{1:t-1}, c_{1:t}) dm_{c_t} p(s_t|s_{t-1}^{[k]}, u_t) ds_t \end{aligned}$$

Applying convolution theorem twice yields the Gaussian,

$$\mathcal{N}\left(\hat{z}_t, \underbrace{G_s P_t G_s^T + G_m \Sigma_{c_t,t-1} G_m^T + R_t}_{L_t}\right) \tag{B-13}$$

The importance weight can be calculated from the Gaussian equation,

$$w_t^{[k]} = |2\pi L_t|^{0.5} \cdot \exp\left(\frac{1}{2}\left(z_t - \hat{z}_t\right)^T L_t^{-1}\left(z_t - \hat{z}_t\right)\right) \tag{B-14}$$

Appendix C

Impact-based SLAM Algorithm

The pseudo-code for impact-based SLAM using particle filter is shown in Algorithm 4. The input to the algorithm are the particle set Y_{t-1} at time $t-1$, measurement z_t, motion noise covariance P_t, measurement noise covariance R_t, control input u_t and minimum correspondence likelihood p_0. The flowchart describing the algorithm is shown in Figure 3-1.

For data association step in the algorithm, computing the most likely landmark is linear in the landmark size along with a matrix inverse computation, scaling $\mathcal{O}(n^3)$, n being the number of dimensions. To improve the computational efficiency, simple geometric constraints apart from orthogonality and corridor width can be taken into account. The landmarks which are not in the view of robot's heading can be pruned from association since it will yield a very low likelihood to the current measurement. Similarly for computing data association for features within the same landmark, certain features are geometrically ordered with respect to the current robot pose. For example, certain feature or wall face of the landmark cannot be viewed from other side of wall.

In the geometric update step, the constraints are incorporated as measurements into the state estimates using an additional EKF update step [29], [16]. The constraints are modelled as a hyperplane of the form,

$$Cx = b + \epsilon \tag{C-1}$$

where C and b model the rectilinear constraints such as orthogonality. The constraints are incorporated with a non-zero covariance $E(\epsilon\epsilon^T)$ between neighbouring landmarks to avoid inconsistency in the SLAM solution. At the end of iterations, the constraints are added with zero covariance for all the landmarks or in other words, it is assumed to be certain that all the landmarks in the map must follow a rectilinear structure.

The histogram update step is carried out using discrete Bayes filter or the simple update procedure (as mentioned in Section 2-4-3). A Gaussian distribution or triangular distribution (used in simulations) is used for updating the histogram followed by normalization if needed. The same holds for adding negative information to the histogram.

Algorithm 4 Impact-based SLAM using particle filter

Input: Y_{t-1}, z_t, P_t, R_t, u_t, p_0
$Y_t = \emptyset$
if measurement is not available at t **then**
    for each particle $k \in Y_t$ **do**
        calculate proposal $s_t^{[k]} \sim \mathcal{N}(s_t | s_{t-1}, u_t)$
        landmark estimates unchanged $\mu_{i,c_t}^{[k]} = \mu_{i,c_{t-1}}^{[k]}$, $\Sigma_{i,c_t}^{[k]} = \Sigma_{i,c_{t-1}}^{[k]}$
        append $\left\langle s_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle$ to Y_t
else
    for each particle $k \in Y_t$ **do**
        for each $c_t \in N$ **do**
            compute equations B-10 and B-11 for given c_t and z_t
            calculate modified proposal $s_{t,c_t}^{[k]} \sim \mathcal{N}\left(s_t; \mu_{s_t,c_t}^{[k]}, \Sigma_{s_t,c_t}^{[k]}\right)$
            calculate likelihood for c_t: $d_{c_t}^{[k]} = |2\pi Z_{t,c_t}|^{0.5} \exp\left(-0.5 \cdot (z_t - \hat{z}_t)^T Z_{t,c_t}^{-1} (z_t - \hat{z}_t)\right)$
        Compute most likely landmark $\hat{c}_t$ ▷ data association
        if all likelihoods less than p_0 **then** ▷ initialize landmark
            compute mean and covariance from equations 3-23 and 3-24
            append estimate to particle $Y_{t-1}^{[k]}$
            EKF update with neighbouring landmarks $\phi_i - \phi_j = n \cdot \frac{\pi}{2}$
            create new histogram with finite support
            update histogram information using discrete Bayes filter
            new particle weight $w_t^{[k]} = p_0$
        else ▷ update landmark
            $N = N + 1$
            check for landmark features if orientations are different
            EKF update step for $\hat{c}_t$ as in equations 3-15 to 3-18
            for each $c_t \neq \hat{c}_t$ **do**
                $\mu_{i,c_t}^{[k]} = \mu_{i,c_{t-1}}^{[k]}$, $\Sigma_{i,c_t}^{[k]} = \Sigma_{i,c_{t-1}}^{[k]}$
            check geometric constraints with neighbours
            EKF update step with measurement $\phi_i - \phi_j = n \cdot \frac{\pi}{2}$
            update histogram using discrete Bayes filter
            compute new particle weight $w_t^{[k]}$ using equation B-14
        prune histogram from consecutive poses
        append $\left\langle s_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle$ to Y_t
    if $N_{\text{eff}} < N/2$ **then** ▷ Adaptive resampling
        resample with probability $\propto w_t^{[k]}$
return Y_t

Bibliography

[1] P. Agarwal, W. Burgard, and C. Stachniss, "Survey of Geodetic Mapping Methods: Geodetic Approaches to Mapping and the Relationship to Graph-Based SLAM," *IEEE Robotics & Automation Magazine*, vol. 21, no. 3, pp. 63–80, 2014.

[2] 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 Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[3] M. Golombek, R. Cook, T. Economou, W. Folkner, A. Haldemann, P. Kallemeyn, J. M. Knudsen, R. Manning, H. Moore, and T. Parker, "Overview of the mars pathfinder mission and assessment of landing site predictions," *Science*, vol. 278, no. 5344, pp. 1743–1748, 1997.

[4] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 4270–4275, IEEE, 2003.

[5] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, pp. 695–701, IEEE, 2002.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[7] T. Bailey and H. Durrant-Whyte, "Simultaneous Localization And Mapping (slam): Part ii," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[8] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A Factored Solution to the Simultaneous Localization And Mapping Problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598, 2002.

[9] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot, "Consistency of the EKF-SLAM algorithm," in *Proceedings of the IEEE Conference on Intelligent Robots and Systems*, pp. 3562–3568, 2006.

[10] G. P. Huang, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based Consistent EKF Estimators for Multi-Robot Cooperative Localization," *Autonomous Robots*, vol. 30, no. 1, pp. 99–122, 2011.

[11] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *Journal of field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[12] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization And Mapping that Provably Converges," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, (Acapulco, Mexico), IJCAI, 2003.

[13] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[14] P. Jensfelt, "Approaches to mobile robot localization in indoor environments," in *PhD thesis, Signal, Sensors and Systems (S3), Royal Institute of Technology, SE-100 44*, Citeseer, 2001.

[15] S. J. Julier, "The stability of covariance inflation methods for slam," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2749–2754, 2003.

[16] P. Newman, *On the Structure and Solution of the Simultaneous Localisation And Map building problem*. PhD thesis, University of Sydney, 1999.

[17] C. Fox, M. Evans, M. Pearson, and T. Prescott, "Tactile SLAM with a biomimetic whiskered robot," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 4925–4930, 2012.

[18] C. W. Fox, M. H. Evans, M. J. Pearson, and T. J. Prescott, "Towards hierarchical blackboard mapping on a whiskered robot," *Robotics and Autonomous Systems*, vol. 60, no. 11, pp. 1356–1366, 2012.

[19] H. Durrant-Whyte and T. Bailey, "Simultaneous Localization And Mapping: Part I The Essential Algorithms," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[20] S. J. Julier and J. K. Uhlmann, "A Counter Example to the Theory of Simultaneous Localization And Map building," in *Proceedings of the IEEE Conference on Robotics and Automation*, vol. 4, pp. 4238–4243, 2001.

[21] A. J. Cooper, *A comparison of data association techniques for simultaneous localization and mapping*. PhD thesis, Massachusetts Institute of Technology, 2005.

[22] S. J. Julier and J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[23] J. A. Castellanos, R. Martinez-Cantin, J. D. Tardós, and J. Neira, "Robocentric map joining: Improving the consistency of ekf-slam," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 21–29, 2007.

[24] S. J. Julier and J. K. Uhlmann, "Using covariance intersection for slam," *Journal of Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 3–20, 2007.

[25] P. Jensfelt and S. Kristensen, "Active global localization for a mobile robot using multiple hypothesis tracking," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 5, pp. 748–760, 2001.

[26] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2432–2437, 2005.

[27] "Sphero-ROS SDK." http://mmwise.github.io/sphero_ros/. Accessed: 2015-08-13.

[28] "Sphero Robotic ball." http://www.sphero.com/. Accessed: 2015-08-10.

[29] D. Rodriguez-Losada, F. Matia, L. Pedraza, A. Jimenez, and R. Galan, "Consistency of SLAM-EKF algorithms for Indoor Environments," *Journal of Intelligent and Robotic Systems*, vol. 50, no. 4, pp. 375–397, 2007.

[30] K. R. Beevers, *Mapping with Limited Sensing*. PhD thesis, Rensselaer Polytechnic Institute, 2007.

[31] S. Bruno and O. El Hamzaoui, "tinyslam: A slam algorithm in less than 200 lines c-language program," in *Proceedings of the IEEE International Conference on Control Automation Robotics and Vision*, 2010.

[32] M. Montemerlo, *FastSLAM: A factored solution to the simultaneous localization and mapping problem*. PhD thesis, Robotics Institute, Carnegie Mellon University, 2002.

Glossary

List of Acronyms

SLAM	Simultaneous Localization And Mapping
AI	Artificial Intelligence
ICRA	IEEE Conference on Robotics and Automation
IMU	Inertial Measurement Unit
EKF	Extended Kalman Filter
MCMC	Markov Chain Monte Carlo
RBPF	Rao-Blackwellized Particle Filter
DBN	Dynamic Bayesian Network
ML	Maximum Likelihood
JCBB	Joint Compatibility Branch and Bound
FSM	Finite State Machine
SIR	Sequential Importance Resampling
CI	Covariance Intersection
SDK	Software Development Kit

List of Symbols

α	Robot heading control
μ_t	Mean of state estimate at time t
$\nabla(\cdot)$	Jacobian of a function
$\psi_r^{[k]}$	Orientation of particle pose
ψ_l	Orientation of landmark
θ	Robot orientation
$(x_r^{[k]}, y_r^{[k]})$	XY- position of k^{th} particle pose
(x_l, y_l)	XY-position of landmark
a_x	Rate of acceleration along the X axis
a_y	Rate of acceleration along the Y axis
$c_{1:t}$	Time history of data associations
$f(\cdot)$	State transition model
$G_{m_{l,t}}$	Jacobian of measurement model over landmark pose
$G_{s_t^{[k]}}$	Jacobian of measurement model over particle pose
$h(\cdot)$	Observation model
K_t	Kalman gain at time t
m	Map of an environment
N	Number of landmarks
P_t	Covariance of state estimate at time t
Q_t	Process noise covariance at time t
R_t	Observation noise covariance
S_t	Innovation covariance at time t
s_t	Robot pose at time t
$S_{l,t}$	Innovation covariance
t	Continuous time index
$u_{1:t}$	Time history of control inputs
v	Robot velocity control
v_t	Observation noise
w_t	Process noise
$z_{1:t}$	Time history of observations