Car park mapping with simultaneous localisation and mapping (SLAM)

Christopher Tay

Car Park Mapping with Simultaneous Localization and Mapping (SLAM)

Tay Meng Keat, Christopher

Report for MR2
September 2004

Supervised by Ceric Pradalier and Christian Laugier

Projet done with team E-Motion,
Laboratory GRAphique, VIsion, Robotique (GRAVIR)
INRIA Rhne-Alpes
ZIRST, 655 av. de l'Europe
38330 Montbonnot St. Martin
FRANCE

Abstract

One of the challenges in the automation of vehicles takes place in the car park. Automatic vehicles must be able to localize itself accurately and build a map of the car park in real time. This capability will be able to extend the range of possible applications including the tracking of car park lots availability, automatic navigation and automatic parking to name a few.

In this project, we took on the task of localizing an automatic vehicle and building a map of the car park in real time. This project takes place within the car park of INRIA Rhone-Alpes on the CyCab vehicle with a Sick laser range scanner. A key feature is that it works only with laser scanners to retrieve the position and orientations of vehicles in the car park. With the detected vehicles as landmarks, CyCab performs a localization of itself and build a map of the car park at the same time. This problem is commonly known in the literature as Simultaneous Localization and Mapping(SLAM).

The detection of vehicles is based on the work of a previous DEA student (Lorieux) [11] and extensions were proposed to increase the reliability of vehicle detection. The SLAM algorithm chosen is the FastSLAM algorithm. The FastSLAM algorithm is adapted within this context. However, FastSLAM only gives a set of hypotheses. Hence, a map construction method is proposed to merge the different hypotheses together to form one single final map.

Experiments with real data were conducted and it is able to perform the tasks required. The experiments also revealed weaknesses in the system and possible approaches and directions for further research is suggested.

Contents

List of Figures

Chapter 1

Introduction

Mobile robots have shown significant promise for remote exploration and to replace humans whenever possible in daily mundane tasks, allowing us to explore in humanly inaccessible places and to help us in improving our lives such that it is easier and more comfortable. To be able to function intelligently, it is imperative to have autonomous capabilities such that it is able to move around and make decisions without human intervention. The autonomy also requires the mobile robot to possess the capability of dealing with the complexities of interfacing with the real world and making sense of it.

At least one of the primary obstacles to making intelligent mobile robots useful and reliable is the uncertainty of robot positioning. After all, mobile robots cannot operate effectively if they do not know where they are. Accurate positioning knowledge is necessary to create high-resolution maps and accomplish repeatable, accurate path following needed for high-level deliberative behavior such as systematically searching or patrolling an area.

With current technology, it is now possible to use Global Positioning Systems (GPS) to be able to position itself. The limitation of GPS systems is its relatively low resolution and because of that, it is difficult to obtain accurate results.

Moreover, there are situations which does not permit the use of GPS systems such as in indoor environments, during a cloudy day, underwater or even in extraterrestrial planets. The process of positioning itself and creating a map of its environment at the same time is formally known as *Simultaneous Localization And Mapping (SLAM)*. However there is a chicken-or-egg problem in SLAM. Uncertainty in the mobile robot's position makes it difficult to pin down the location of the landmarks. To lower the uncertainty of the mobile robot's position, it needs to perceive the landmarks and know the landmarks' position with low uncertainty. This chicken-or-egg relationship between localization and mapping is a consequence of imperfection in the

machines we make, namely odometry and sensory errors. As the robot moves, estimation of its own position is corrupted by noise in its odometry. Similarly, its perception of the real world is far from ideal and highly corrupted. However, there is a correlation between the position and orientation of the robot with what it sees and the error is systematic. The seminal paper by Smith, Self and Cheese [17] demonstrated that by exploiting the correlation, a solution is mathematically possible.

1.1 Background and Problem Description

The project addresses the problem of creating a map of the cars in a car park. Testing is done in the car park of INRIA Rhone Alpes. The physical setup consists of a bi-steerable electrically powered vehicle: CyCab. It has 4 electric motors powering the wheels. And it looks like an advanced version of a golf car. There is also a laser sensor that is mounted in the front (refer to 1.1.1).

Figure 1.1: CyCab at INRIA in red and blue

1.1.1 Sensory Perception

CyCab is currently only tested in the car park of INRIA Rhone Alpes. CyCab itself is equipped with a laser scanner by the name of *LMS(Laser Measurement System)-219*. LMS-219 is situated at about 50cm from the ground, on the front side of CyCab. It scans the front side of CyCab with a view of 180 degrees. The range of the LMS-219 scanner is about 20 meters. LMS-219 works by rotating about an axis and emits a laser. The laser will bounce off a surface and will be received by LMS-219. The time delay is then used to calculate the distance. As such, the LMS-219 data is susceptible to noise subjected to the material makeup of the surface reflecting the laser.

1.2 Problem Description

In this project, we attempt to create a map of the car park with CyCab. The map of the car park will contain the positions and orientations of the different vehicles that exist in the car park. Such a map can serve as a reference to indicate obstacle positions and dimensions. Furthermore, it can indicate the state of the parking lots in the car park, and be used in more high level situations like automatic parking.

The general idea is that using only the LMS-219 laser data and no artificial or predefined landmarks, CyCab will navigate the car park autonomously while generating a map of its environment. One of the requirements for sucessful navigation is the ability to find its own position accurately in the car park. With no predefined landmarks to serve as reference points, a wrong estimate of its own configuration will also lead to a map which contains errors.

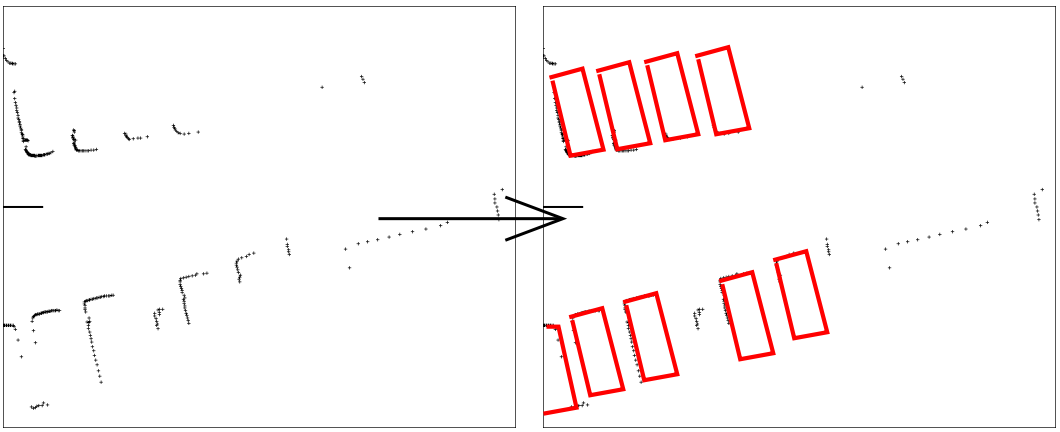

Figure 1.2: Process of obtaining the configuration of vehicles based on data from LMS-219 scan data

The input and output is illustrated in 1.2. The set of impact points from LMS-219 is displayed on the left of the diagram. CyCab will then move around the surroundings and it builds the map of the car park as shown on the figure's right.

In order to simplify the problem, certain assumptions are made to impose a reasonable restriction. As an intitial assumption, the car park should be static. But it is expected that the problem and its solution presented can be extended to include the few moving vehicles in the car park. Furthermore, the sizes of the vehicles does not vary too widely. For example, chances of finding a limousine parked in a car park mostly populated with normal sized cars is low. Car parks mostly contain vehicles belonging to the same class or size. The design of a car park containing cars of various sizes is awkward and most of the time, there are different car parks for vehicles belonging to different classes.

Within the context of this problem, CyCab will be explicitly referred to and the other vehicles other than CyCab itself will be referred to generally as vehicles.

It is also assumed that the vehicles are polygonal and takes the shape of a rectangle and in the car park, there are no other objects which takes the shape of the vehcicles.

1.3 Approach Adopted

While CyCab is travelling around the car park, scanning the environment, CyCab continuously reads in odometric and laser data iteratively. At each stage of the iteration, it estimates its own position and orientation in the form of (x, y, θ) and creates a map of the car park all in the world frame of reference. The origin of the world frame of reference is take from the initial position of CyCab. The map is then represented as a set of tuples, each containing the position and orientation of the vehicles detected. CyCab hypothesizes the configuration of the vehicles in the surrounding based on the laser scan inputs from LMS-219 only.

Common SLAM techniques normally makes use of artificial landmarks or perform alignment and matching of the scan maps as a form of landmark. However, in our approach, we used the hypothesized vehicles as landmarks in helping us to localize and map the car park, all without using any additional and artificial external aids. However, as it will be further discussed in the later chapters, there are also disadvantages to it.

1.4 Organisation of Report

- **Overview** (Chapter 2) gives a general appreciation of the entire system in general and shows the main components involved in obtaining the map from raw laser data. The main components involved, namely extraction of vehicles, SLAM and map construction will be presented in order after the next chapter (state of the art).
- **State Of The Art** (Chapter 3) aims to give a survey of the literature available on the various SLAM techniques and the justification for choosing the selected SLAM method
- **Extraction Of Vehicles** (Chapter 4) describe in detail the processes required to obtain vehicle hypothese from raw laser scan data. The extractoin of vehicles is built on top of a previous student's work. His work will be presented briefly. Our proposal on its extensions are described in detail towards the end of the chapter.
- **SLAM** (Chapter 5) explains the FastSLAM technique which is used to localize CyCab and obtain the landmark(vehicles) positions and directions. The chapter tries to present the

crucial concepts in FastSLAM without getting too involved with the mathematical details

- **Map Construction** (Chapter 6) presents our approach in merging the different hypotheses from FastSLAM into one single final map
- **Results and Future Directions** (Chapter 7) are then presented. The results are presented visually with screenshots of the program and generally divides into 3 sections. One for each of the 3 components described in the Overview (Chapter 2). Our opinion on the future possibilities are also presented
- **Conclusion** (Chapter 8) summarises the work done and contributions. It also describes in brief the problems encountered and recapitulates on points mentioned in previous chapter but attempts to give a view of the relationship between the various components now that we have understood the project's strength and limitations better

Chapter 2

Overview

The mechanism of the entire system can be broken down into three fundamental portions, *vehicle detection*, the *simultaneous localization and mapping(slam)* and the *map construction* . CyCab is provided with two kinds of raw data, the laser scans from LMS-219 and CyCab's odometric data. Figure (fig. 2.1) shows the block diagram of the different stages and its interactions:

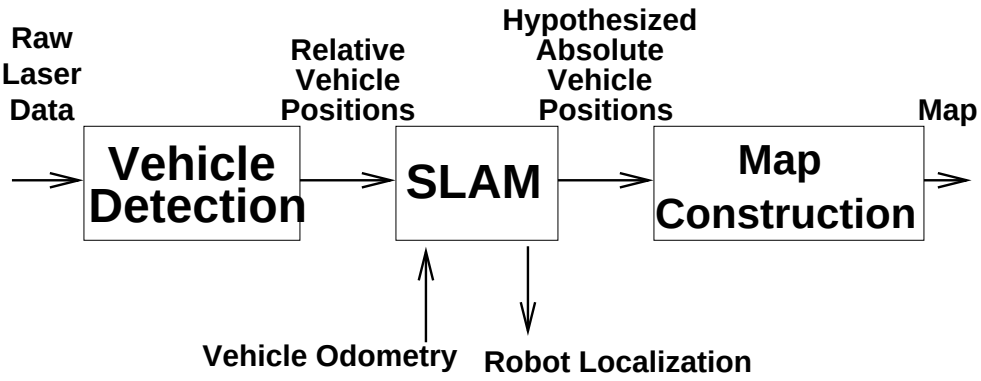

Figure 2.1: Overview showing the mapping process

1. **Vehicle Detection**: With only raw laser scan data, vehicle detection makes hypotheses about the positions and orientation of vehicle in the car park.

2. **SLAM**: Coupled with information about odometry of CyCab, SLAM tries to make sense of its sensorial perception and calculates its own configuration such that it matches the sensorial perception as accurately as possible while closely conforming to the odometry data. With its own configuration, SLAM can then provide information about the absolute configuration of the vehicles, which are configurations with respect to real world coordinates. A point to note about the absolute configuration is that the real world frame of reference is with respect to the starting position of CyCab itself.

3. **Map Construction**: As will be explained in later chapters (chap. 5), the method of slam used (FastSLAM) outputs a set of different hypotheses. Hence, a map construction module is required to merge the information from the different hypotheses to obtain the final map

2.1 Vehicle Detection

Laser scan data is fed into the vehicle detection module where it tries to make the different hypothesis of the configuration of the vehicles. Which means, given a laser scan, it tries to deduce the vehicle positions and orientations as accurately as possible. The hypothesized vehicle configurations are with respect to CyCab's frame of reference. These information are passed on to the SLAM module. The vehicle configurations with respect to CyCab's frame of reference forms the sensorial perception of CyCab.

Vehicle detection requires the processing of the raw laser scan data from LMS-219. This is because the information provided by LMS-219 is a set of impact points in the form of angle and distance (α, d).

In order to create hypothesis of possible car positions, CyCab must be able to see the two sides of the vehicles. A LMS-219 laser scanner requires line of sight and thus is able to see at most two sides of a vehicle at any one time. To get lines indicating possible sides of the vehicles, a four stage sequential process is used. These stages were previously developped by the last DEA student, Jean Lorieux [11], at team E-Motion

1. **Clustering** is performed to group the points close together, which most probably indicates an object.

2. **Segmentation** is performed on the groups of points which possibly forms part of the contour of the object represented by the cluster. In this way, we can recover the lines which forms the shape as indicated by the laser impacts as closely as possible.

3. A **Histogram** is constructed to indicate the possibility of the existence of the vehicles at various configurations. The peak values are extracted from the histogram as it indicates the most probably positions of the different vehicles.

4. The **polygon of visibility** is constructed and it represents a region where we are sure that there are no obstacles present. The segments from segmentation forms the boundary of a polygon of what is known as the polygon of visibility. As the peaks detected in the histogram might probably contain a number false positives, the polygon of visibility can be used to filter off a number of spurious hypothesis.

However under real circumstances (e.g. vehicles which are not exactly rectangular), it becomes a little unreliable and produces still an unacceptable amount of false positives. The concept of edge filtering is proposed to extract the essential and relevant edges integral to the vehicle. A metric to measure how much the edges fit the edges of a car is introduced. This metric is calculated after edge filtering. The details can be further explained in section 4.6

2.2 Simultaneous Localization And Mapping (SLAM)

SLAM tries to make sense of its sensorial perception and calculates its own configuration such that it matches the sensorial perception as accurately as possible while closely conforming to its own odometry data. With its own configuration, SLAM can then help provide information about the absolute configuration of the vehicles, which are configurations with respect to real world coordinates. A point to note about the absolute configuration is that the real world frame of reference is taken to be the starting position of CyCab itself.

2.2.1 Principles of SLAM

To have an idea of the basic input and output parameters involved in SLAM, we consider the simple scenario. We have a robot equipped with a sensor capable of detecting its environment features (such as the LMS) placed in a room. The pose of the robot at time t shall be represented as s_t. In this case, the configuration of CyCab (x, y, θ) is represented as its x-y position in the plane of the floor it is travelling on and its orientation. The trajectory of CyCab since the start can be written as:

$$s^t = \{s_1 \dots s_t\} \tag{2.1}$$

The robot moves through the environment given the control u_t at time t. The set of controls shall be represented as

$$u^t = \{u_1 \dots u_t\} \tag{2.2}$$

A moving robot collects information about its own motion like from odometric sensors in the wheels or the set of control commands given to it. As it moves, it makes observations z_t about its environments. With the set of observations collected since its start:

$$z^t = \{z_1 \dots z_t\} \tag{2.3}$$

Most of the time, the robot makes observations specifically for any of the N landmarks at locations $\{\theta_1 \dots \theta_N\}$. And the set of all landmarks collectively forms a map represented by Θ. Each observation provides information on one of the landmarks θ_n, with n representing the identity of the landmark observed. The association of the observation z_t with a certain

landmark θ_n is known as *data association*. In practice, it is very difficult to single out individual landmarks from the laser scan because they are normally not unique. Hence algorithms are needed to perform the mapping of observations to the corresponding landmarks. The mapping of observation z_t to landmark θ_n is represented by n_t, where n_t holds the value of the landmark number. For example, $n_1 = 10$ would mean that the observation z_1 at time $t = 1$ corresponds to the landmark θ_{10}. The set of data associations made over time is represented in the same notation:

$$n^t = \{n_1 \dots n_t\} \tag{2.4}$$

For mathematical simplicity, it is assumed that at any one time, there can only be either a measurement z_t or a control u_t as inputs to the robot. But in practice, they can be treated one after another sequentially.

The objective of SLAM, represented in its probabilistic form, is to find the distribution of the SLAM posterior:

$$p(\Theta, s^t \mid z^t, u^t, n^t) \tag{2.5}$$

Equation 2.5 expresses the joint distribution of the map (which are made up of landmarks) and robot path given that the robot makes certain landmark observations, is given controls to navigate around and data association information to map observations to landmarks.

2.2.2 Principles of FastSLAM

The SLAM algorithm implemented in this project is FastSLAM by Montemerlo et. al [12] [13]. Naturally, FastSLAM is one of the methods used to solve the problem of SLAM and falls into the subset of problems as described in the section 2.2.1 and it tries to find the SLAM posterior 2.5. FastSLAM is based on the simple observation that given knowledge of the path of the robot, the locations of the landmarks of the robots are conditionally independent.

FastSLAM is implemented as a particle filter. A particle filter consists of a set of 'points' or particles with a weight that is assigned to each particle. The set of particles and weights forms a representative of a probabilistic distribution; In the context of FastSLAM, the posterior distribution.

In FastSLAM, each particle contains its own hypothesis on the robot's path and the landmark locations. Weights are assigned according to how much its observations of landmark positions corresponds to its own belief on the corresponding landmark positions. In each time step, the distribution is refined by a process of resampling with probabilities proportional to the particle weights. If the hypothesis on the path of the robot is more accurate, it will have a good match in its observations and the beliefs of landmark locations and hence, receive more weights. The

resampling process will then be able to slowly sieve out the more accurate hypothesis. The problem of data association is also resolved inherently. A wrong data association will give a bad match in observations and belief of landmark locations which results in the hypothesis receiving low weights. FastSLAM is essentially about maintaining a set of hypothesis and eliminating the inaccurate ones as more measurements are made.

2.2.3 Application within context

As stated in the problem description (sect. 1.2), the purpose is to build a map of the car park without any predefined reference. But CyCab needs to be able to localize its own position accurately in order to give the correct locations of the vehicles in the car park. If we take the vehicles in the car park as landmarks, we are then able to use SLAM to solve the problem of localizing CyCab, yet build a map of the car park at the same time. The method of FastSLAM is then applied to the problem of localizing CyCab and car par map building. In this context, the state of the robot refers to the state of CyCab. Observations made are made specifically to vehicles in the car park. The map that is to be constructed at the end is a set of vehicle positions.

2.3 Final Map Construction

The reason for a map construction module is due to the fact that FastSLAM is manifested as a particle filter. With different particles representing different configurations of CyCab in the configuration distribution, the particles also contains diverse sets of landmark estimates. Hence, the need to merge the different landmark estimates together to form the final map.

In general the map construction works by assigning identities (ID) to landmarks grouped according to the same observation. An observation might involve several different IDs. If this is the case, then we have to choose the ID group that maximizes the landmark posterior of landmarks belonging to the certain ID group.

Chapter 3

State Of The Art

Virtually all state-of-the-art algorithms in mapping are probabilistic. Probabilistic in the sense that the robot model and sensor model are represented with a distribution and maps are constructed using probabilistic inference methods. The reason for its popularity and power lies in the fact that the uncertainties in measurement and movement can be characterised by probabilistic distributions and such methods take into account the varying degree the noise affects the system as a whole. Since the 1990s, probabilistic methods became extremely popular. This chapter will present the general SLAM methods adopted and in particular, the different ways of SLAM due to its difference in interepreting laser scan data.

3.1 General SLAM Methods

3.1.1 EKF Approaches

Seminal papers written by Smith, Self and Cheeseman [17] [18] introduced a statistical framework to solve the problem of SLAM. Since then, the process of creating a map using robots have been given the name Simultaneous Localization And Mapping (SLAM). It proposed to use Extended Kalman Filters to estimate the SLAM posterior.

EKF represents SLAM as a high dimensional multivariate Gaussian with mean indicating the state of the robot and landmarks, and covariance indicating the pairwise correlation within the set of robot and landmarks. EKF is an extension of Kalman Filters (KF) [8] to non-linear systems by replacing the non linear motion and sensor models of the robot with its linearized version.

The main disadvantages of using EKF is that of quadratic complexity and single hypothesis

data association. As the number of landmarks increases, the size of the EKF covariance matrix grows quadratically. Furthermore, the EKF maintains only a single hypothesis on the landmark positions. The hypothesis is normally chosen using methods of maximum likelihood. If the probability of an observation is too low, it is further evaluated as a new landmark. However, if it makes a wrong association, it will never be able to recover from it and the EKF will diverge.

3.1.2 EM Approaches

An alternative to using KF in SLAM is to use *expectation maximization (EM)* [21] [23]. EM is a statistical algorithm originally proposed by Dempster, Laird and Rubin [5]. EM is one of the better techniques used to solve the problem of SLAM in cases of large-scale cyclic environment even when the features look alike and cannot be distinguished easily. The EM algorithm iterates two steps continuously:

1. **Expectation step (E-step)** is used to calculate the posterior over robot poses.
2. **Maximization step (M-step)** in which the most likely map given the robot pose calculated in the E-step. This step employs the method of maximum likelihood.

As it iterates, it gets increasingly accurate maps. In fact, it is performing a hill climbing search in the map space and it has a need to process the map several times. EM hence cannot generate maps incrementally and not suitable for real time applications.

There are some solutions which tried to emulate the EM approach but with much faster computational times. Common ones include the *incremental maximum likelihood method* [24] [25]. It can be interpreted as EM without the E-step. Basically, it incrementally builds a map with each observation but without keeping track of residual uncertainty. And because of the inability to track residual uncertainty, it does not handle cyclic environments well.

3.1.3 Submapping

Due to the weakness of EKF methods mentioned previously (sect. 3.1.1), it is too expensive computationally to implement such methods for large maps. As a result, extensive research has been done on SLAM algorithms to approximate EKF algorithms, but on larger environments. The main weakness of EKF methods stem from the covariance matrix, which contains every pairwise correlation between state variables.

Since an observation on a landmark will have a very small effect on landmarks far away, which renders a portion of the EKF covariance matrix a very small value, methods have been developed

to exploit this fact by decomposing the global map into smaller submap regions. Postponement [9] and Compressed Extended Kalman Filter are examples of such techniques. These techniques delay the update of local map information into the global map, assuming that the robot stays in the region of a submap for some time. Such techniques achieves the same accuracy as EKF methods, just that the computation time is reduced by a constant factor.

Other maps divide the submaps more distinctively. One of the more promising approach is the ATLAS [3] framework. ATLAS is a general framework used to cope with large maps. In ATLAS, the global map is divided into different local submaps. The relationships between the submaps are represented as graphs with each node containing the submap information and edges representing the transformation and uncertainty between the two adjacent submaps. In each submap, the uncertainties are modelled locally. The transformation and uncertainties between any two arbitrary submaps are calculated using dijkstra's algorithm (a commonly used graph algorithm to find the shortest distance between two arbitrary nodes of a graph) with the uncertainty information contained in edges as costs.

3.2 SLAM with laser scanners

Of particular interest within the project is to look at other techniques using laser scanners like the LMS to perform SLAM. There were different methods of interpreting the laser scan data. The two main interpretation of scan data is either to perform SLAM by looking for certain specific features or to perform alignment of the scan data.

In a bid to be independent from landmarks, localization by scan matching were proposed. One of the more popular methods is by Thrun, Burgard and Fox [22] where the scans alignment is done by computing its occupancy grid maps from previous measurements. Maps are updated using incremental likelihood maximization [5]. Such scan matching methods is even applied to the case FastSLAM [12] [13] in generating maps using only raw laser range measurements and it is capable of closing large loops in maps. Another approach proposed by Eliazar and Parr [2] which performs almost the same thing and uses a grid as well. But it is different from traditional occupancy grid maps in that each grid stores a balanced tree which contains the hypothesis's belief in the occupation of the grid in question. The essential difference here is that in traditional FastSLAM like approach to SLAM with the use of particle filters, most methods associate a map to each particle, and this method associates particles to a single map. And thus, the maintanance of the maps and poses are more efficient.

Another class of scan matching methods is to recover a set of short line segments to represent the environment and to perform matching using the lines. However, such methods are relatively

less used compared to those using occupancy grid maps. An assumption that it requires is that its environment needs to have objects containing lines and such. It probably would not work well under natural environments like parks.

3.3 Data Association

In SLAM, data associations are rarely observable as most of the time we are unable to extract the exact identity of a landmark given an observation because landmarks are often indistinguishable. Data associations are of paramount importance in SLAM because the only means of localization is by referencing to landmarks observed. A wrong association of observation to landmarks will cause serious localization errors and frequently unrecoverable. But if uncertainty in landmark positions is low relative to average distance between landmarks, simple data association heuristics can work rather well.

One common simple heuristic is the maximum likelihood. It computes a set of different probabilities for landmark identities given an observation. The landmark generating the highest value will be the most likely data association. And if the value is below some fixed threshold, the observation is considered a new landmark.

However, such a heuristic does not work well for high uncertainties in landmark position. Sometimes, the heuristic might make a wrong data association and for cases like the EKF, a wrong data association will cause divergence of SLAM. A number of sophisticated methods have been proposed to deal with more intelligent and reliable data association (sect. 3.3.1, 3.3.2).

3.3.1 Joint Compatibility Branch and Bound

In cases of multiple observations which occur frequently in practice, maximum likelihood approach (sect 3.3) will treat each data association independently. In reality, the data associations for the different observations are related because the uncertainties comes frequently from uncertainty of the robot pose. The maximum likelihood approach might sometimes yield the same landmark for different associations and the problem of mutual exclusion is ignored.

Neira and Tardos [15] [19] proposed an algorithm called the Joint Compatibility Branch and Bound (JCBB). It constructs an interpretation tree which consists of a tree of possible joint correspondences. Different hypotheses are computed by calculating the probability of the observations occuring together. However, the computational cost is high as it considers an exponential number of hypotheses. To avoid traversing the entire tree of possibilities, intelligent pruning can be performed by removing impossible hypotheses.

3.3.2 Local Map Sequencing

Tardos et al [20] [19] used a technique known as Local Map Sequencing to build maps of indoor environments using sonar data. The sonar inputs are processed by two hough transforms looking for line segments and corners. The hough transforms makes data associations by taking sensor readings from different robot poses and suggested the use of RANSAC [7] to vote for the correct interpretation of data associations.

3.4 FastSLAM

The FastSLAM algoritm [12] [13] is a recent SLAM approach that has made progress in the field. It is based on the Rao-Blackwellized particle filter [6] proposed by Murphy in 1999. He proposes that the filter sample robot poses and track the position of a fixed number of pretermined landmarks using Kalman Filters. This is based on the fact that landmark positions are conditionally independent give the robot pose.

Key features of the FastSLAM algorithm is that it maintains a multiple hypotheses track on the robot pose and its environment. And because of the multiple hypotheses, it is able to tolerate a certain amount of uncertainty and robustness. Furthermore, it does not require explicit loop closing heuristics and is able to run in real time with only a few hundred hypotheses.

The choise of FastSLAM over other SLAM algorithms is mainly due to reasons of computational efficiency, the potential to execute in real time. It is possible to close loops which is a desirable feature especially within the context of the car park because such car park configurations are common. Furthermore, we do not require submapping algorithms as normally, the car park is not very big.

Chapter 4

Extraction of Vehicles

The extraction of vehicle configuration requires a few stages of processing to make sense of the group of impact points. There exists in the literature methods to extract vehicle configurations from range image sequences by Zhao and Thorpe [26]. Their detection takes place on the highway where there are only cars. But in the car park, the situation is a lot more complex as there exists structures nearby which are not vehicles but trees, pedestrians and we have to deal with problems of occlusion. The method here is similar in spirit to [26]. But our method is more adapted to search for possible vehicle positions in the car park. The various sub components will be briefly explained with detailed elaborations on the pitfalls of the previously developped methods and proposals to overcome them.

The methods described in this section is based on the previous DEA work by Jean Lorieux [11] (chapters 2 and 3). However it does not work very well for more complicated real life data (described in section 4.5 and we will show in section 4.6 the proposed extension to more reliable vehicle extraction.

4.1 Clustering

The aim of clustering is to group the impact points such that impact points within the group represents the sides of the same object. This is possible within the context of the car park because objects in the car park are relatively uncluttered. With the points already grouped, segmentation of lines from the points makes more sense logically. If the points were not grouped, there will exist segments which do not form a side of any object and vastly reduces the accuracy of the vehicle configuration hypotheses. In figure 4.1, the impact points and its segmentation are illustrated. It is quite obvious where the position of the vehicles are. If clustering is not performed, an unwanted extra segment illustrated as a dotted line will appear. Grouping the

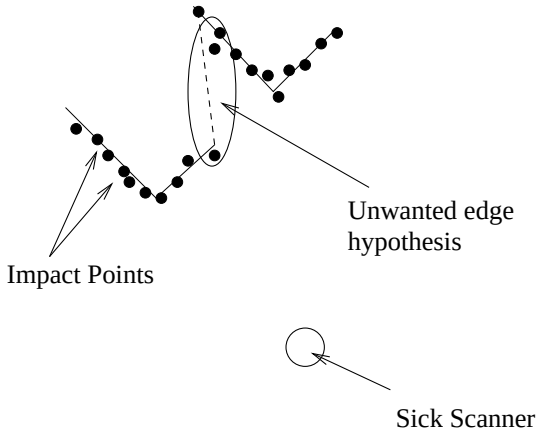

Figure 4.1: Image showing the unwanted segment hypothesis in dotted lines.

points together increases its processing efficiency as there are fewer points to consider each time we perform the segmentation. The clustering is performed in the manner described in the following. For each impact point, it checks if the distance to a current cluster in question is below a certain treshold. The current impact point is added to the cluster if it is below the treshold. If not, a new cluster is simply created. Figure 4.2 illustrates an example of the output of clustering where the relevant group of points indicates the same segment, and that dubious segments are eliminated.

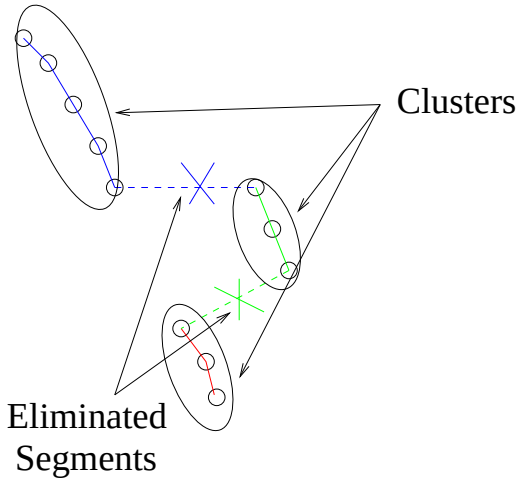

Figure 4.2: An example illustrating the results of clustering

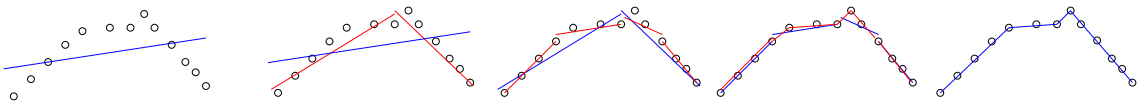

Figure 4.3: The various stages in recursive bisection with linear regression

4.2 Segmentation

The segmentation proposed by J. Lorieux [11] is inspired by standard *split and merge* techniques commonly found in the literature [10] [4] [19]. Such techniques takes a group of points and try to fit a contiguous segment that will approimately fit the set of data points. As the name split and merge implies, segmentation is performed by recursively splitting the group of points into subgroups and perform line fitting. And at each stage, each subsolution to the segmentation will be merged together.

As the group of impact points grouped together do not necessarily form a straight line, a comprehensive segmentation algorithm is required. To do so, the points are segmented recursively. Figure 4.3 illustrates the sequence involved in the recursive segmentation. At each stage, the linear regression of the current group of points is calculated along with the point furthest away from the segment. The group of points is divided evenly into 2 and the same calculations are performed. If the difference between the distance of the parent group and of its subgroup is less than that of a prespecified treshold, the segment belonging to the parent is kept and the recursion stops there. If not, the operation continues in its two sub impact point groups.

4.3 Histogram Construction

From the segments, a 3D histogram is filled with values representing the possible configurations of the hypothesized vehicle which corresponds to the segments, with the 3 axes of the histogram mainly representing the x and y positions of the hypothesized vehicles and their orientations θ. In essence, the value contained in a single cell within the 3D histogram represents the probability of the hypothesized vehicle with configuration (x, y, θ). The values are filled up by considering for each segment, all the possible configurations that is in accord with the segment in question. In figure 4.4, we can see a graphical representation of all the possible configurations with a side of the hypothesized vehicle along the edge that does not contradict the segment in question. For each configuration, the probability of the configuration is added to the value contained in the corresponding histogram position. A more detailed exposition on the calculation can be found in J.Lorieux's report [11].

After the histogram is filled up, it is passed through a treshold filter removing all values

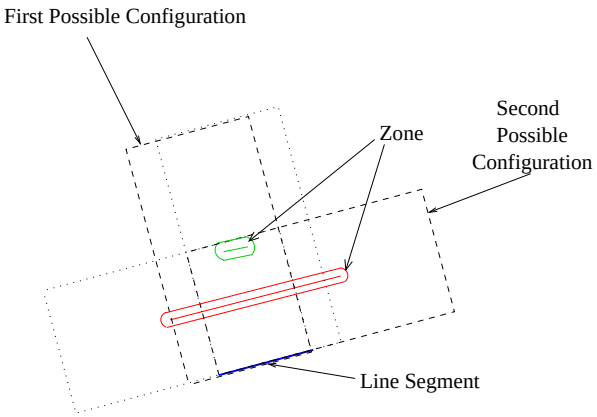

Figure 4.4: Relationship between the edges and the correspoding possible configurations of vehicles such that it is to be filled up in the 3D histogram. The red zone indicates the possible configurations corresponding to the length of the vehicle and the zone in green represents the possible configurations corresponding to the width of the vehicle

below the specified treshold. An example is illustrated in figure 4.5. In this figure, we can see 3 diagrams. The first diagram shows the segments obtained from the laser data scans after applying segmentation (sect 4.2). After segmentation, the histogram is calculated and visualised on the 2D x-y plane as shown in the middle. On the last diagram towards the right, we can see the dark regions of the histogram, which represents higher probability values, showing the positions of hypothesized vehicles with strong probabilities as it correponds to the segments. Regions of strong probability surfaced in the histogram normally takes the shape of lines which corresponds to the segments obtained.

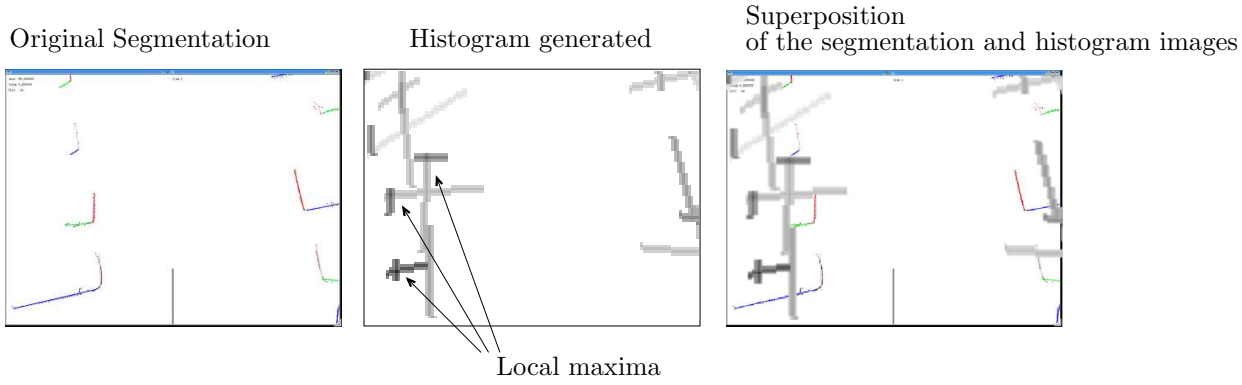

Figure 4.5: The series of figures shows the original segmentation, the histogram generated from the segmentation and the relationship between the segments and histogram by superposing one on top of another

We can see that there are some areas that indicates the most probable configurations of vehicles and they are located at the local maxima of the histogram. To identify the local maxima and group them together, a simple recursion algorithm is performed which is similar to recursive

flood fill algorithms in computer graphics. For each point in the histogram, it is compared with its neighbour points and the point containing the bigger value gets to spread its zone into the weaker points. The ensemble of zone points is then represented by a gaussian.

4.4 Polygon of Visibility

The gaussian mean parameters indicate the position of the hypothesized vehicles. However, it consists of numerous false positives. Filtering the vehicle configurations with the polygon of visibility reduces the number of false positives. The polygon of visibility is a polygon whereby we are sure that the interior of the polygon consists of only free spaces. It is constructed by joining the series of segmentation maps that is obtained as mentioned in section 4.2.

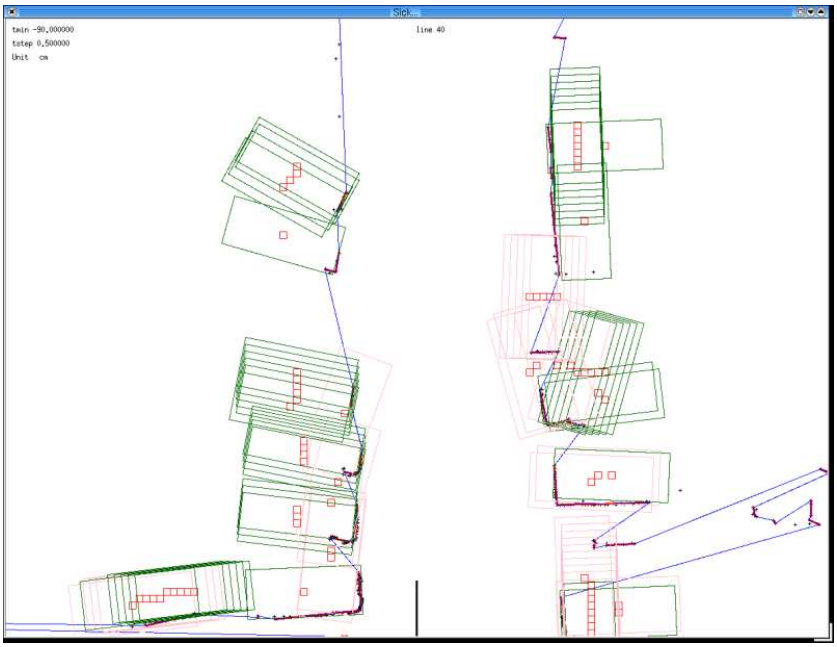

Figure 4.6: Filtering by the polygon of visibility. The polygon in blue indicates the polygon of visibility formed by segmentation. The green vehicles indicate valid positions and the red indicate vehicles eliminated by the filtering.

In the process of filtering, for each vehicle hypothesis, the area of intersection of the vehicle with the polygon of visibility is calculated. If the area of intersection is greater than a certain threshold, it means that the hypothesis contains a relatively large portion of the vehicle which lies in the area of which we are sure there are no obstacles and the hypothesis can be safely eliminated.

4.5 Problems

After a series of experiments with real life data taken from the Sick mounted on Cycab, we found the filtering by polygon of visibility to be not sufficient in dealing with the numerous possible vehicle and segment configurations. The problem encountered can be classified into 2 categories as shown in figures 4.7 and 4.8.

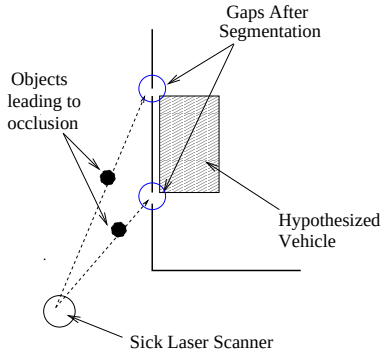

Figure 4.7: First problem category. A wrong vehicle hypothesis due to gaps in segmentation when it is meant to be a long straight segment

The first category involves the case where there are gaps between line segments as indicated in 4.7. Take for example the corner of a building (in fig 4.7). Due to unpredictable laser scan uncertainties or minor occlusions, clustering is not able to group all the impact points belonging to the same wall of the building together. Hence, a broken segment of appropriate length might mislead the program. This occurs rather frequently under numerous different circumstances. A line segment corresponding to a side of the vehicle induces a hypothesis that a vehicle is there when it is not. The hypothesis is not eliminated by the polygon of visibility.

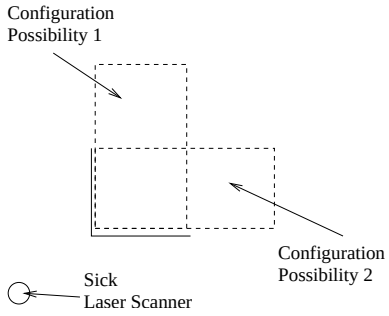

Figure 4.8: Second problem category. A confusion between two possible configurations given two adjacent segments

The second category is one where the length of two adjacent sides of a vehicle renders it impossible to tell one configuration from another when both can fit the line segments (fig. 4.8). Section 4.6 proposes some methods to eliminate such ambiguities.

4.6 Proposed Modifications

In order to eliminate the problems encountered in section 4.5, one has to consider the fundamental question of what in a segmentation map enables us to be sure of a certain vehicle configuration. The definition we have chosen is that to be sure of a certain vehicle configuration, we must verify that the two adjacent sides corresponds more or less to the dimensions of the vehicle.

Two extensions proposed are edge filtering and calculation of vehicle support. The purpose of edge filtering is to choose the appropriate segment from the segments such that it is close enough to the hypothesized vehicle and is reasonable as a side of the hypothesized vehicle. With the segments obtained from the edge filtering, we can efficiently calculate the vehicle support, which is a measure of how well the sides conform to the vehicle hypothesis configuration.

4.6.1 Edge Filtering

To be able to calculate how well the sides conform to a vehicle hypothesis configuration, we need to extract for each hypothesized vehicle, the set of segments relevant to the hypothesized vehicle. And this is what edge filtering attemps to do. But the main aim of edge filtering is to serve as data for the calculation of vehicle support (section 4.6.2).

Edge filtering delves into the domain of geometry. The edges of interests are the edges which lies around the contours of the vehicle. In order to do so, two bounding rectangles are calculated from the vehicle hypothesis configuration with one rectangle a ratio smaller than the original vehicle size and the other a ratio bigger as illustrated in figure 4.9. The two bounding rectangles will then be oriented and positioned in the same manner as the hypothesized vehicle configuration.

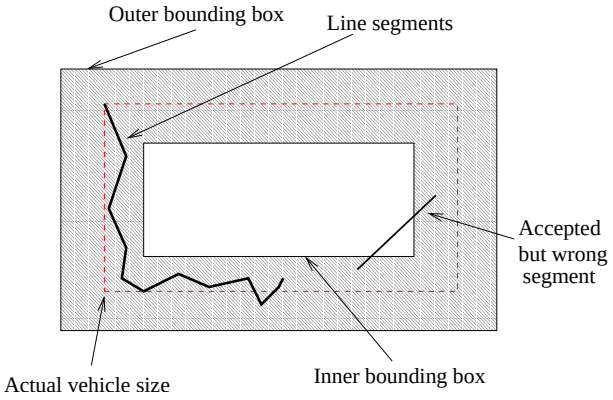

Figure 4.9: Edge filtering with 2 bounding boxes. Shaded area indicates valid area

Line segments are kept if the start and end points of the segment lie within the outer bounding box and outside the inner bounding box which is indicated by the shaded area in figure 4.9. Some might wonder what about the case where both the end points lie within the shaded region but its edge cuts through the inner bounding box. This is highly improbable. The argument is that if this were the case, the vehicle hypothesis would already have been eliminated by filtering using the polygon of visbillity.

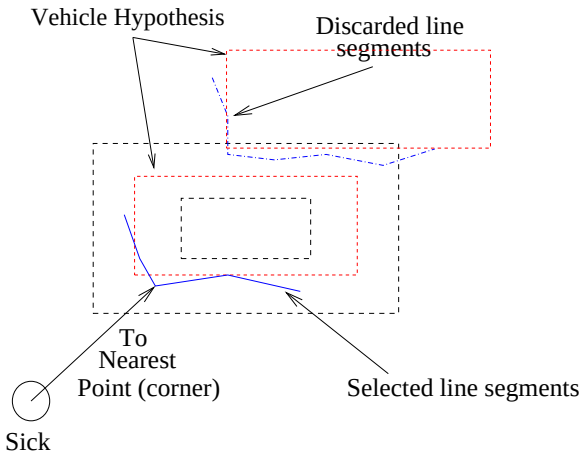

Figure 4.10: Possible segments still lie within the two bounding boxes. However, selecting segments spreading from the corner allows the correct segment to be selected

The problem now is to extract from these set of segments, those that only originate from the corner of the hypothesized vehicle. There is a possiblity for lines to exist on other sides of the vehicle and yet does not belong to the original vehicle hypothesis (fig. 4.10). Hence to extract those segments related to the corner, an algorithm is created that begins with the segment with one of the endpoints nearest to the origin (where Sick is). Starting from this segment, the algorithm starts to grow outward by searching for any segments where any of its endpoints lies sufficiently close to one of the endpoints in the original segment (example in figure 4.11). And this continues till we cannot find a sufficiently close segment.

4.6.2 Vehicle Support Calculation

Although we have a set of segments, we have to find out how well the sides conform to the vehicle configuration. This is because we have to validate that we are able to see both sides of the vehicle adequately and not only one of its sides or portions of either sides which introduces ambiguities and false positives like the errors mentioned in section 4.5. By merely adding up the length of the segments does not validate the vehicle configuration for the segments might just be at one side of the vehicle. According to the proposed definition, the vehicle configuration hypothesis is true if we see the two sides, another metric based on the sum of the magnitude of

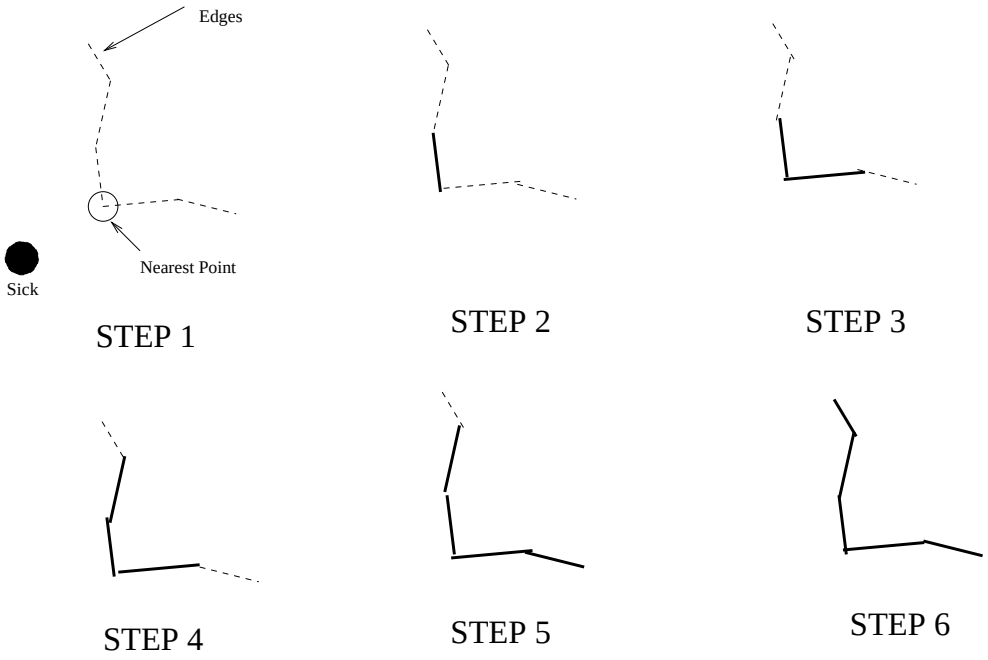

Figure 4.11: A simple example showing the spreading of the edge method. Note here that the ends of the edges need not join together

cross products can be used to evaluate the vehicle support. As we all know:

$$| \overrightarrow{A} \times \overrightarrow{B} | = | \overrightarrow{A} | \, | \overrightarrow{B} | \, sin(\theta) \tag{4.1}$$

By going through each pair of segments and evaluating the equation 4.1 then adding them up, we can obtain the vehicle support. Under the ideal case where there are only two segments perfectly aligned to the edges of the vehicle, the result is a multiplication of the length and width of the vehicle. Hence the equation for calculating the support:

$$support = \forall i, j \sum_{i \neq j} | \, S_i \times S_j \, | \tag{4.2}$$

Suppose that in the ideal case, we have a set of segments $\{A_1 \ldots A_l\}$ that corresponds ideally to the length of the car and another set of segments $\{B_1 \ldots B_l\}$ that corresponds ideally to the width of the car. We have for every segment A_i exactly perpendicular to every segment B_j. And under such ideal conditions,

$$length = \sum_i A_i \tag{4.3}$$

$$width = \sum_j B_j \tag{4.4}$$

The support will be

$$\begin{aligned} & \forall i,j \sum_{i \neq j} \mid S_i \times S_j \mid \\ = \; & \forall i,j \sum_{i \neq j} \mid S_i \mid\mid S_j \mid sin\theta \\ = \; & A_1 \sum_j B_j + A_2 \sum_j B_j + \ldots + A_l \sum_j B_j \\ = \; & \sum_i A_i \sum_j B_j \\ = \; & length * width \end{aligned} \tag{4.5}$$

From equation 4.5, we can see that if there are no segments either in A or in B we have a support of zero. This property is especially usefull in eliminating false positives from the first problem case as stated in section 4.5. Furthermore, it ensures that we have substantial support from either segments in A and B to have a high support value. To determine if a set of segments provides good support for consideration of a vehicle hypothesis to be correct, the value of the support is calculated and if it is higher than a treshold, it is accepted as a vehicle position. For the case of the second problem case (described in section 4.5), when we meet ambiguity, it means that we are unable to decided which orientation the vehicle hypothesis is. The vehicle support is also correspondingly low. Hence the strategy adopted is a conservative one. We wait till we see more of the sides of the vehicles, hence giving it a high support value before we decide that it is a vehicle position.

Intuitively, if we see a large portion of the length but a small portion of the width, the support will give a low value as can be seen from equation 4.5. This is the same when we observe more of the width than the length. Hence it is important to be able to see both sides of the hypothesized vehicle before classifying it as a positive vehicle hypothesis. The enforcement of such rules is all conveniently embodied into a single equation (eqn 4.2).

Chapter 5

Simultaneous Localization And Mapping (SLAM)

In the standard SLAM framework, a mobile robot executes conrols and accumulates observations of the environment, both of which are corrupted by noise. The controls and observations can be put under constraints probabilistically under a suitable noise model. In the beginning, such constraints may be uncertain. But as the mobile robot moves around and makes more observations, it is able to correspondingly adjust the constraints such that the constraints becomes more and more rigid. Ideally, with infinite control and observation inputs, it is able to converge on a solution where the position of the landmark observations and its own position will be correlated.

In the overview (Chapter 3), we have briefly explained the basic principles of SLAM (sect 2.2.1) from a probabilistic point of view. With the same set of variable naming convention used in chapter 3, the aim of SLAM from a probabilistic point of view is to estimate the SLAM posterior:

$$p(\Theta, s^t \mid z^t, u^t, n^t) \tag{5.1}$$

In calculating the posterior 5.1, two basic distributions available to us are the measurement model (eqn. 5.2) and the motion model (eqn. 5.3).

$$p(z_t \mid s_t, \theta, n_t) \tag{5.2}$$

$$p(s_t \mid u_t, s_{t-1}) \tag{5.3}$$

In the project, observations are obtained from the vehicle configuration hypothesis. The odometry information provided by CyCab, is the angle ϕ of the wheel and the distances d_L, d_R travelled by the two rear left and right wheels of CyCab respectively. Such odometry information from

CyCab provides sufficient information in predicting the motion of CyCab in the next time step with small uncertainty, under the hypothesis that the curvature is locally constant.

5.1 FastSLAM

In this project, the SLAM method used is FastSLAM [13] [12]. FastSLAM is based on the fundamental observation that given the true path of the robot, the estimates of the various landmark positions will be conditionally independent. This statement is best expressed by the decomposition of the FastSLAM posterior as follows:

$$p(\Theta, s^t \mid z^t, u^t, n^t) = p(s^t \mid z^t, u^t, n^t) \prod_{1}^{N} p(\theta_n \mid s^t, z^t, u^t, n^t) \tag{5.4}$$

Where the first term of the decomposition of eqn 5.4,

$$p(s^t \mid z^t, u^t, n^t) \tag{5.5}$$

represents the path posterior of a mobile robot. The second term of the decomposition of eqn 5.4 ,

$$p(\theta_n \mid s^t, z^t, u^t, n^t) \tag{5.6}$$

refers to the posterior distribution of landmark θ_n. Each individual probabilistic distribution in the decomposed SLAM posterior can be updated using an estimator recursively. Such a factorization was first introduced by Murphy and Russel in 1999 [14]. One has to note also that this decomposition is an exact decomposition and not an approximation.

5.1.1 Particle filters in FastSLAM

The conditional independence of the landmark positions given the path of the robot allows us to decompose the SLAM posterior. In reality, we do not know the true path of the robot. If we knew the true path of the robot, we would no longer need to solve the problem of SLAM because the robot would be localized with certainty. However, this decomposition will be possible if we are able to maintain a set of robot path hypothesis and update the landmarks conditioned on the hypothetical path. And by validating the robot path with observations, we are able to eliminate the less probable paths.

Particle Structure

Particle filter is used to represent the different hypotheses, with each particle containing a single hypothesis. In fact, the set of particles represent arbitrary probability distributions by

concentrating particles in regions of high probability and having few or no particles in regions of low probability. In the case of FastSLAM, the distribution represented by the particles is the path posterior of the robot. Within each particle filter, there are N landmark estimators in the form of Extended Kalman Filters (EKF) which are conditionally dependent on the path of the robot represented by the particle itself. Because landmarks are estimated using EKFs, each landmark is represented as a gaussian with mean μ and covariance Σ. The total number of kalman filters required will hence be $N.M$, N kalman filters for each for the M particles. As a result, each fastslam m^{th} particle at time t is of the form:

$$S_t^{[m]} = s^{t,[m]}, \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, \ldots, \mu_{N,t}^{[m]}, \Sigma_{N,t}^{[m]} \tag{5.7}$$

Particle Distribution

The initial thought of representing the path posterior (eqn 5.5) is by sampling from the path posterior distribution directly. But, we are unable to do so because the path posterior distribution is what we are trying to find out as well. Since we are unable to sample from the SLAM posterior directly because it will be contradictory to do so, we have to use a proposal distribution. Each particle can be assigned a weight with the help of a proposal distribution according to the ratio:

$$weight = \frac{target\ distribution}{proposal\ distribution} \tag{5.8}$$

A new set of unweighted particles is then drawn from the current set of weighted particles in proportion to the weights calculated using the proposal and target distribution (eqn 5.8). This process is known as *sampling importance resampling (SIR)* [1].

A good criteria for a proposal distribution is that the proposal distribution will have to be as close as possible to the posterior. In the first proposal of FastSLAM (a.k.a FastSLAM 1.0), the proposal distribution chosen is of the form:

$$p(s^t \mid z^{t-1}, u_t, n^{t-1}) \tag{5.9}$$

With this proposal distribution, it does not consider the measurement z_t at time t. In cases where the motion uncertainty is relatively large compared to the measurement uncertainty, sampled poses of the robot mostly falls into areas of low measurement likelihood. Under this situation, most of the particles will receive negligible weights. And by the next resampling phase, the few particles with non-negligible weights receives a significantly higher weightage. This situation creates a rather ineffecient sampling scheme and is wasteful with the samples, and at the same time accelerate the rate of sample impoverishment. The incorporation of measurement requires only a change in the proposal distribution which results in a proposal distribution of the following:

$$p(s_t \mid s^{t-1}, u_t, n^t, z_t) \tag{5.10}$$

5.1.2 FastSLAM Algorithm

Like in standard SLAM algorithms, FastSLAM performs the update of landmarks and the robot's position incrementally while repeating the process for each new observation of landmarks made. Generally, FastSLAM takes place in 4 basic stages:

1. For each particle, given a control, a new pose is sampled
2. Update the landmark positions using EKFs of observed landmarks for each particle
3. Calculate importance weight
4. Perform importance resampling

Sampling a new pose

In FastSLAM 1.0, poses are drawn from the standard motion model $p(s_t \mid u_t, s_{t-1})$. And in FastSLAM 2.0, measurements are incorporated in and drawing poses from the new proposal distribution is not evident. Poses in FastSLAM 2.0 are sampled according to the distribution:

$$s_t^{[m]} \sim p(s_t \mid S^{t-1,[m]}, u^t, z^t, n^t)) \tag{5.11}$$

A decomposition of the proposal distribution yields the following:

$$\begin{aligned} p(s_t \mid s^{t-1,[m]}, u^t, z^t, n^t) \quad \propto \quad & \eta^{[m]} \int p(z_t \mid \theta_{n_t}, s_t, n_t) p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1}) d\theta_{n_t} \times \\ & p(s_t \mid s_{t-1}^{[m]}, u_t) \end{aligned} \tag{5.12}$$

With reference to equation 5.12, the first term, $p(z_t \mid \theta_{n_t}, s_t, n_t)$, refers to the probability distribution for measurement z_t from state s_t given that we know the data association and landmark. The probability distribution for measurement comes from the measurement model in equation 5.2. The second term, $p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1})$, refers to the landmark position distribution, which is essentially a gaussian distribution from the last EKF landmark udpate. The third term, $p(s_t \mid s_{t-1}^{[m]}, u_t)$, refers to the motion model as already presented in equation 5.3.

Sampling from this distribution is not possible. This is due to the fact that the measurement model is not linear. In the context of the project, a measurement is given by:

$$g(s_t, \theta_{n_t}) = \begin{bmatrix} r(s_t, \theta_{n_t}) \\ \phi(s_t, \theta_{n_t}) \end{bmatrix} \tag{5.13}$$

The measurement model in this case is a polar coordinate reference (r, ϕ) to landmark observations in the LMS' frame of reference (refer to fig 5.1). With the linearization of the measurement model using taylor's expansion, the integral in equation 5.12 can be reduced to a closed form and can be sampled from directly. Such an approximation is similar to the EKF approach (For details please refer to [13]).

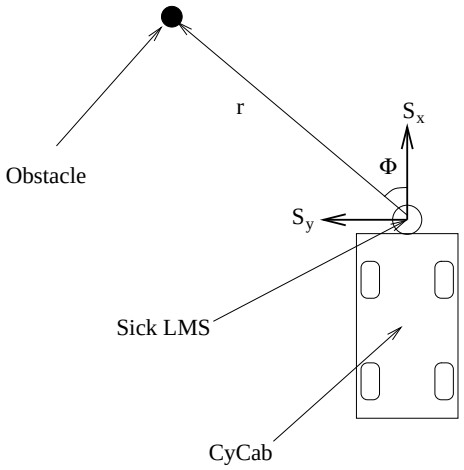

Figure 5.1: Illustration of the measurement model with S_x and S_y denoting Sick LMS' frame of reference

Updating landmarks

With more landmark observations, we are getting more information on the landmark positions. In FastSLAM, the different landmarks observed are processed one after another using EKF. The update of landmark position estimates is performed according to the posterior:

$$p(\theta_{n_t} \mid s^{t,[m]}, n^t, z^t) \quad \propto \quad \eta p(z_t \mid \theta_{n_t}, s_t^m, n_t) p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1}) \tag{5.14}$$

In the decomposition as shown in equation 5.14, the first term $p(z_t \mid \theta_{n_t}, s_t^m, n_t)$ is the non linear measurement model while the second term refers to the landmark distribution in the form of a gaussian from the last landmark update. The non linear measurement model is again linearized using taylor's expansion. And this sort of EKF styled updates is in the same spirit as in calculating the proposal distribution (sect. 5.1.2). (For mathematical details, refer to [13]).

Importance weighting and resampling

Equation 5.8 shows the general calculation to obtain the weight values necessary for the resampling stage. Under the asymptotic assumption that $s^{t-1,[m]} \sim p(s^{t-1,[m]} \mid z^{t-1}, u^{t-1}, n^{t-1})$. The new proposal distribution hence becomes:

$$p(s^{t-1,[m]} \mid z^{t-1}, u^{t-1}, n^{t-1}) p(s_t^{[m]} \mid s^{t-1,[m]}, z^t, u^t, n^t) \tag{5.15}$$

Going by equation 5.8,

$$\begin{aligned} w_t^{[m]} &= \frac{target\ distribution}{proposal\ distribution} \\ &= \frac{p(s^{t,[m]} \mid z^t, u^t, n^t}{p(s^{t-1,[m]} \mid z^{t-1}, u^{t-1}, n^{t-1}) p(s_t^{[m]} \mid s^{t-1,[m]}, z^t, u^t, n^t)} \\ &= \dots \\ &= \eta \int \int p(z_t \mid \theta_{n_t}, s_t, n_t) p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, u^{t-1}, n^{t-1}) d\theta_{n_t} \times \\ &\quad p(s_t \mid s^{t-1,[m]}, u^t) ds_t \end{aligned} \tag{5.16}$$

Similarly, the measurement model, $p(z_t \mid \theta_{n_t}, s_t, n_t)$, can be linearized with taylor's expansion and both the landmark posterior ($p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, u^{t-1}, n^{t-1})$) and motion model ($p(s_t \mid s^{t-1,[m]}, u^t)$) are gaussian, the solution is in the form of a gaussian as well. (Details in [13]).

5.1.3 Incorporating Multiple Observations

There are moments where a mobile robot can observe two landmarks at a single time. In the context of the project with Sick LMS, the laser scan data produces two detection of vehicles. Traditional methods of treating multiple observations is to treat the observations as if the observations arrived one after another.

However, in the case of FastSLAM, each observation will cause an EKF measurement update as shown in section 5.1.2 when calculating the proposal distribution. Each iteration of the EKF update will reduce the uncertainty in the proposal distribution. Because of this property, the addition of new landmarks should be placed after the update of landmarks in order to achieve the smallest possible uncertaintty in the proposal distribution. In this way, the uncertainty of the new landmark will be smaller when compared to if the landmark had been added earlier.

5.2 FastSLAM Characteristics

5.2.1 Data Assocation Properties in SLAM

FastSLAM maintains multiple hypothesis on path and on data association. For each observation the robot makes, the robot performs the data association by choosing the landmark which maximizes the probability of how well a measurement fits the expected measurement:

$$\widehat{n_t} = \underset{n_t}{argmax}\ p(z_t \mid s^{t,[m]}, z^{t-1}, u^t, n^t) \tag{5.17}$$

Data association is performed for each particle seperately, and independently from other particles. Hence, each particle contains not only its own hypothesis on the path, but on the data association

as well. Such an approach is known as per-particle maximum likelihood(ML) data association. Data association according to equation 5.17 is a form of the *Nearest Neighbour (NN)* approach. Although NN might cause single hypothesis SLAM algorithms such as the EKF to diverge, it works a lot better for FastSLAM. One of the main reason why it works is that motion uncertainty induces high data association ambiguity. However, there will be some particles that draws poses that is approximately that of the true robot. These particles receive correct data association and explain the observations well.

It is because of this mechanism of drawing the more consistent hypothesis form a set of hypotheses that frees FastSLAM from the need to have very extensive data association capabilities which are often computationally costly.

Chapter 6

Map Construction

In this project, we have used the method of FastSLAM as described in chapter 5 to obtain a set of hypothesis containing the path of the robot (hence its state at current time) and its L landmarks within each hypothesis m. A hypothesis therefore consists of:

$$hypothesis\, m = \{s_t^{[m]}, \theta_1^{[m]} \dots \theta_L^{[m]}\} \tag{6.1}$$

But in the end, we need a single map that will be used for other applications. The different hypotheses provides no single fix on robot path and landmark positions such that it is useful. The aim of map construction is thus to produce the final map by combining the different hypotheses together. The map which is made of N landmarks is denoted as:

$$\Theta = \{\theta_1 \dots \theta_N\} \tag{6.2}$$

Within the context of this project, the map is a set of vehicle positions found in the car park.

6.1 Landmark Tagging

Before we begin describing the method used to combine the different hypotheses to obtain the final combined map, and for reasons of clarity, we have to define the concept of landmark tagging.

The idea behind landmark tagging is to assign an identification number (ID) to each physical landmark observed. Each of the different hypotheses contains its own belief of landmark locations. By tagging landmark beliefs of every hypothesis with a unique ID, we are able to associate the hypothesized landmark position to a physical observation.

Going by this description, each time an observation is encountered, each hypothesis will make data associations of observations to landmarks which is performed during SLAM. If the

hypothesis associates the observation to a new landmark, a newly generated ID will be assigned. Otherwise, it will be an observation of a previous landmark. In this case, the old ID will be retained. By doing so, we are able to uniquely identify each of the landmarks of all the hypotheses under a common context.

6.2 Process Flow

The construction of the final map is performed over two stages sequentially:

1. The first stage calculates the posterior of the landmark for every hypothesis.
2. In the second stage, the landmark ID is used to group the set of landmark hypothesis together. However, different hypotheses makes the association between landmarks and observations differently and there is the possibility of having several IDs associated to a single observation. The second stage resolves this conflict and produces the final landmark to be included or updated on the map.

6.2.1 Landmark Posterior

In the estimation of landmark positions, the landmark posterior was proposed by Schulz and Bugard [16] is of the form:

$$p(\theta \mid z) \tag{6.3}$$

In our case, we evaluate this posterior for every landmark that was believed to be observed for every hypothesis. In FastSLAM, each and every landmark posterior within the context of a single hypothesis is given by the expression

$$p(\theta_{n_t} \mid s^t, z^t, u^t, n^t) \tag{6.4}$$

This expression is evaluated with EKF during execution of FastSLAM. The final distribution obtained from the EKF is a gaussian of the following form:

$$p(\theta_{n_t} \mid s^t, z^t, u^t, n^t) \sim N(\mu_{\theta_{n_t}}, \Sigma_{\theta_{n_t}}) \tag{6.5}$$

Within the framework of FastSLAM, the reduction to obtain the posterior $p(\theta \mid z)$ is performed as follows:

$$\int_{s^{t,[m]}} p(\theta_{n_t} \mid s^t, z^t, u^t, n^t)\, p(s^t \mid z^t, u^t, n^t) ds^{t,[m]} = \int_{s^{t,[m]}} p(\theta_{n_t} s^t \mid z^t, u^t n^t) ds^{t,[m]} \tag{6.6}$$

$$= p(\theta_{n_t} \mid z^t n^t) \tag{6.7}$$

The inverse measurement prediction model $g^{-1}(s_t, z_t)$ gives the expected landmark location given the state of the robot and the landmark observation by the robot. And since, the particle filter in FastSLAM represents the distribution $p(s^t \mid z^t, u^t, n^t)$ (note that this distribution is found in the second term of the integral of equation 6.6), we are able to calculate equation 6.7 in the context of particle filters by:

$$\begin{aligned} p(\theta_{n_t} \mid z^t n^t) &= \int_{s^{t,[m]}} p(\theta_{n_t} \mid s^t, z^t, u^t, n^t)\, p(s^t \mid z^t, u^t, n^t) ds^{t,[m]} \\ &= \sum_{s_t^{[m]}} N(g^{-1}(s_t^{[m]}, z_t); \mu_{\theta_{n_t}}, \Sigma_{\theta_{n_t}}) \end{aligned} \tag{6.8}$$

6.2.2 Conflict Resolving

As explained previously (section 6.2), there is the possibility of having several IDs associated to a single observation z_t. This group of M IDs is represented as a set I consisting of elements of unique IDs i_m:

$$I = \{i_1 \dots i_M\} \tag{6.9}$$

For each i_m of the group I, there exist a set, L_m, of landmarks observed by the different hypotheses with ID i_m. The conflict is resolved by choosing one of the IDs to represent the landmark for the single observation z_t. The chosen ID is the ID $i_{\widehat{m}}$ which gives the maximum value of the sum of all landmark posteriors belonging to L_m:

$$\widehat{m} = \underset{m}{argmax} \sum_{\theta_{n_t} \in L_m} p(\theta_{n_t} \mid s^t, z^t, u^t, n^t) \tag{6.10}$$

Hence the landmark with ID $i_{\widehat{m}}$ is added onto the final map and the rest of the landmarks, $I - i_{\widehat{m}}$ will be removed from the final map if it already exist in the final map. The landmark position is computed by the average of the landmarks in the set $L_{\widehat{m}}$.

The rationale for this is that the set of hypotheses represents the distribution of the robot path. But there will however be some hypothesis which strays off and might interpret an observation for another wrong landmark ID and are not consistent with most of the other hypotheses. Hence such strayed off hypotheses is prohibited from adding landmarks to the final map.

There is a case to take note of. This is when the bulk of the hypotheses is relatively varied and there is no significant mass of consistency of hypotheses. This happens when there is no observations for some time and due to odometry uncertainty, the hypotheses in FastSLAM will start to spread out and report a larger number of IDs after an observation. This case can be easily detected as the maximum of the sum of landmark posteriors in equation 6.10 will have smaller value. The small value also imply high uncertainty and inconsistencies among the hypotheses. Only values above a predefined treshold is allowed to add landmarks to the final map.

Chapter 7

Experimental Results and Future Directions

7.1 Vehicle Detection

The original vehicle detection method was developped by J.Lorieux [11] (chapters 2, 3). We proposed the method of edge filtering followed by vehicle support filtering. Our method is able to reduce the number of false positives. The two problematic cases as mentioned in section 4.5 performs satisfactorily. Figure 7.1 shows the position of vehicles in colour red, signifying that these hypothese are accepted. Taking note especially of the left vehicle in figure 7.2 which is a wrong hypothesis but it is accepted. But after applying edge and vehicle support filtering, the hypothesis is rejected (rejected hypothesis in cyan).

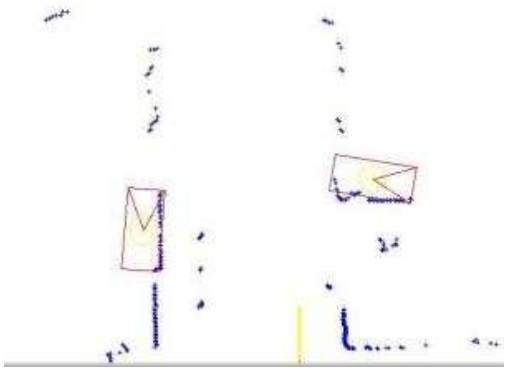

Figure 7.1: Vehicle hypotheses accepted without edge and vehicle support filtering

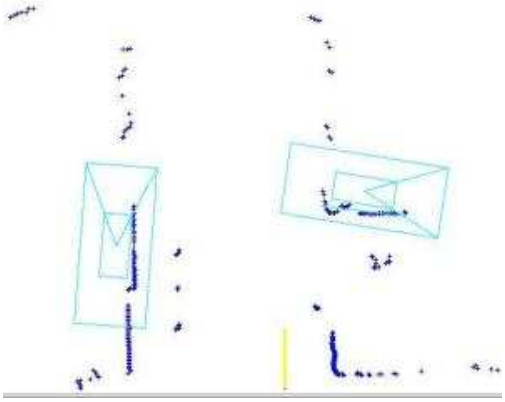

Figure 7.2: Vehicle hypotheses rejected by edge and vehicle support filtering

In figures 7.3 and 7.4 are examples of the second problem case (mentioned in section 4.5)

where there are two possible configurations of the vehicle given two edges but one of them is obviously wrong. The vertically aligned vehicle hypothesis in figure 7.3 was accepted because no edge filtering is applied. But with the application of edge and vehicle support filtering, the vertically aligned vehicle which is obviously in the wrong orientation is rejected (colour shown in cyan).

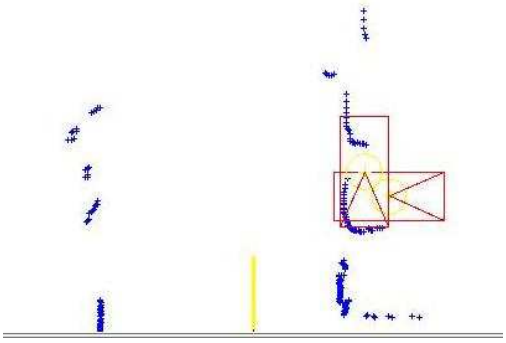

Figure 7.3: Vertically aligned vehicle hypotheses accepted without edge and vehicle support filtering

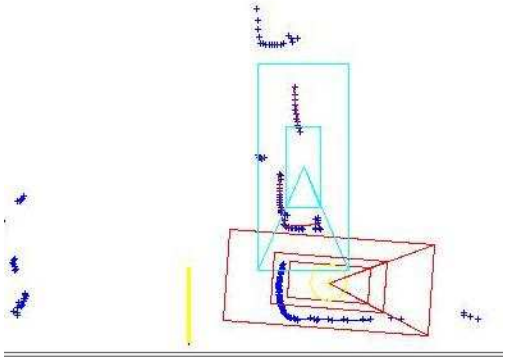

Figure 7.4: Vertically aligned vehicle hypotheses rejected by edge and vehicle support filtering

However, in the course of applying the proposed methods of edge filtering (sect. 4.6.1) and vehicle support filtering (sect. 4.6.2), some of the potential vehicle hypotheses were eliminated. This shows that the method of edge filtering with vehicle support calculation validates the vehicle hypotheses more strongly, but at the expense of several potential hypotheses. Figures 7.1 and 7.2 contains an example. In figure 7.1, the vehicle hypothesis on the right is originally accepted when no edge and vehicle support filtering is applied. But after edge and vehicle support filtering, the vehicle hypothesis is rejected along with the other vehicle hypothesis illustrated in the diagram(fig. 7.2). This is due to the fact that the criteria for edge filtering and vehicle support filtering is stringent and the vehicle hypothesis fails one of the two criterias at least.

Due to the numerous possibilities that could occur, our method of vehicle detection still is not entirely foolproof. Take for example in figure 7.5. In this figure, it can be seen that due to occlusions, the segments are cut off in a very suggestive manner that there validates the vehicle hypothesis in red when in fact, it is just a corner of the building.

7.2 SLAM

Several tests were ran on data taken from CyCab fitted with Sick LMS. From the experiments, it is obvious that odometric uncertainties were significantly larger than uncertainties with Sick

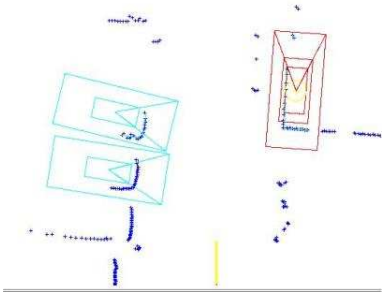

Figure 7.5: A wrongly made hypothesis despite edge and vehicle support filtering

LMS, especially when CyCab is turning. The current data association method used is the per-particle ML data association method.

Because of the fact that odometric uncertainty is larger than observation uncertainty, CyCab has a constant need to be able to observe landmarks to keep itself in check. If CyCab is unable to take frequent landmark references when it turns for some time, the state estimate of the hypotheses will go off track, along with the hypotheses of future landmark positions. Take for example a scenario (fig 7.6). The boxes in green represents vehicle hypotheses while the boxes in black represents CyCab state hypotheses. From figure 7.7, we can see that finding landmark positions are not of a big problem. From the path traced out in blue, it is evident that there are no big angle turns.

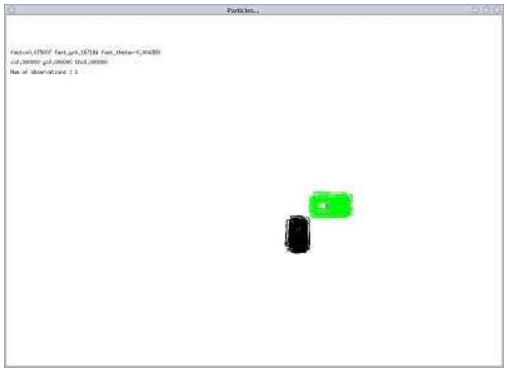

Figure 7.6: Start of navigation. Boxes in green are vehicle hypotheses while boxes in black are CyCab hypotheses

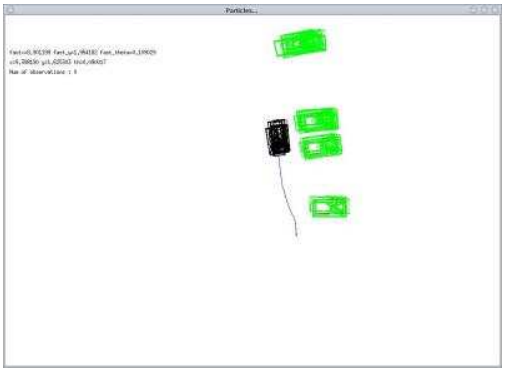

Figure 7.7: Still works because no large turns

A turn is performed (fig 7.8), due to odometry errors, the true position of CyCab is not consistent with the CyCab state hypotheses in black (fig 7.9). Data association errors have been marked out in circles and the arrows indicating the original group of vehicle hypotheses (which are in the wrong position due to odometric errors) should be associated with the group of vehicle

hypotheses at the arrow head.

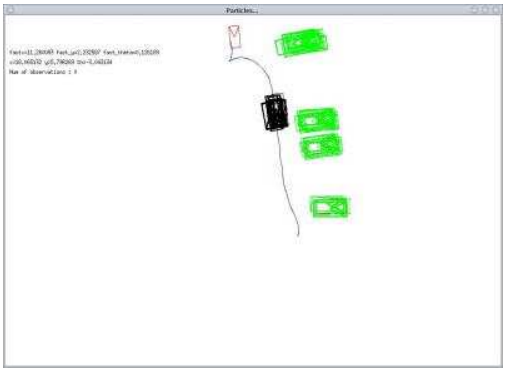

Figure 7.8: CyCab has started to make turns. The box in red indicates the mean from the motion model because there are no observations and thus CyCab state hypotheses are not updated.

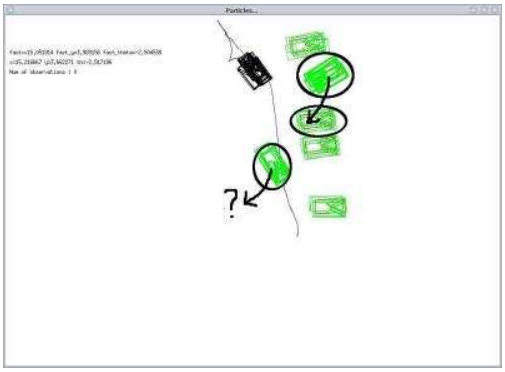

Figure 7.9: Wrong association of data

A possible solution to the problem is to have as many particles as possible to cover all the possible paths. However, it will be impractical as it takes too much computational time. This also illustrates the importance of diversity in the particles. With diversity, then it is possible to cover more hypotheses. However, a possible solution would also be to improve the detection of vehicles such that FastSLAM is able to make more observations constantly.

7.3 Map Construction

Map construction runs successfully on tests performed so far. This is mainly due to the fact that the tests were run within the context of the project. Vehicles as landmarks do not make very cluttered landmarks and hence, the ambiguity is low. Therefore, it remains to be seen the performance of the map construction method in more cluttered environments.

In figure 7.10, we can see the CyCab hypotheses in black and the various landmark hypotheses in green. And it is in figure 7.10 what the map construction module perceives. After which the map construction module constructs the map correctly and which can be easily verified manually.

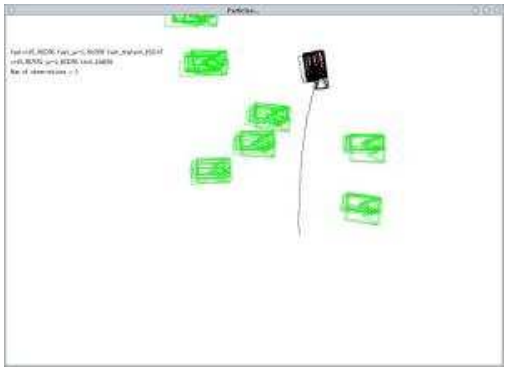

Figure 7.10: Diagram of CyCab hypotheses in black and landmark hypotheses in green

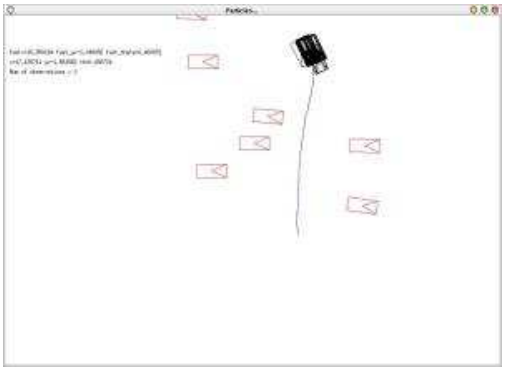

Figure 7.11: Diagram of the map itself with landmark hypotheses in red and CyCab hypotheses in black

7.3.1 Suggested Future Work

Vehicle Detection

To combat these cases, a more general approach needs to be adopted. Edge filtering and vehicle support only caters to a subset of all the possible configurations for laser impact maps. Furthermore, a laser scanner only approach is limited in the sense that it can only retrieve 2D information of at most half a side of an object. Constraints such as assumptions about the shape of vehicles being unique has to be imposed as a result.

There has been a lack of literature on the detection of landmarks using both laser scanners and vision systems. A possible solution would be to have both laser scanner and vision systems to detect vehicles in the car park. Laser scanner provides 2D depth information while vision systems, which do not have the 2D plane information (unless stero vision is applied) , can provide optical information. The laser scanner and vision system can process their inputs independently and perform a fusion of information or cross validation to achieve more accurate and reliable landmark detection.

SLAM

The current slam method assumes static landmarks. Which means that an assumption is made that vehicles are not moving in the car park. This assumption is not a very realistic assumption as there are frequently more than one moving vehicles in a car park at the same time. A possible extension to FastSLAM would be to use particle filters as well to perform the tracking of vehicles.

7.3.2 Map Construction

As was stated earlier in section 7.3, experimental conditions is not enough to really test the map construction module out. A more rigorous study of the proposed map construction method requires more experimental testing under more cluttered environmental conditions.

Chapter 8

Conclusion

A survey of the available literature concerning object extraction from laser scan data and on SLAM has been presented in chapter 3. The idea of performing SLAM using laser scanners is a very popular idea and different methods of processing scan data has been proposed. Most of them used laser scan data to perform alignment of scans or they look for specific predetermined landmarks. However, our approach is a little different. Like the methods using scan alignment, we do not need predetermined landmarks. But we are able to recover object based information which are vehicles parked in the car park and use them as landmarks.

The original work by Lorieux [11] on the detection of vehicle using laser scan data only has several problems (sect. 4.5). It mainly lies with false detection of vehicles. The correct detection of vehicles is important as it affects data association in SLAM which in turn affects SLAM itself. The concepts of Edge filtering (sect. 4.6.1) and vehicle support filtering (sect. 4.6.2) was proposed to reduce the number of false positives. The two proposed methods did reduce the number of false positives, but at the cost of throwing away potential hypotheses. Future directions in the fusion of information from vision (using cameras) and laser scanners is suggested as they are complementary and provide more information together. Furthermore, there has been relatively research in vision with laser scanner based systems.

The method of FastSLAM (chap. 5) has been chosen to perform localization. The main reason is out of concerns of efficiency. One of the requirements is to be able to run in real time on CyCab. Furthermore, FastSLAM represents the state of the art in recent years on SLAM research. The main problems encountered was that there has been too few landmarks (vehicles) detected to provide information to FastSLAM constantly to localize itself correctly. In order to be able to cope with such difficulties, more particles are required to explore more hypotheses. However, with an increase in the number of particles, it will increase the computational run time of FastSLAM rendering it ineffective for real time applications. A solution would be to improve

the detection of vehicles like using the suggested approach of fusion of sensor information from both camers and laser scanners. The current assumption that FastSLAM makes is that the environment is static. Car parks are unfortunately not static most of the time. Hence, a reasonable step next is to perform SLAM while tracking moving landmarks.

A map construction (chap. 6) method is proposed to fuse the hypotheses generated from FastSLAM. The approach proposed groups landmarks according to observations. With this grouping, we are then able to evaluate the most probable landmark with the landmark posteriors. The map construction method works well with current data. This is mainly due to the fact that landmarks are not cluttered vehicles in the car park limits the minimal distance between landmarks. For a more comprehensive evaluation and validation of the map construction method proposed, more challenging data can be applied to test its limits.

On the whole, the system is able to localize CyCab and build a map of the car park. From laser scanner data input to output of a map is currently bordering real time. But in this project, emphasis is placed on the study and experimentation rather than building a reliable, optimised version. Accuracy is an area to be improved and there are still some problems mentioned previously in this chapter which demands attention. However, the system has the potential to be able to sucessfully localize CyCab and build an accurate map of the car park in real time.

Bibliography

[1] *Bayesian Statistics 3*. Oxford University Press, 1988.

[2] Predetermined Landmarks Austin. Dp-slam: Fast, robust simultaneous localization and mapping without.

[3] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2003.

[4] R. Chatila. Representation issues. Technical report, Stockholm, KTH, 2002.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society B*, 39:1–38, 1977.

[6] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filters for dynamic bayes net.

[7] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach*, 24(6):381–395, 1981.

[8] R. E. Kalman. A new approach to linear filtering and prediction problems. In *Trans. ASME, Journal of Basic Engineering*, volume 82, pages 35–45, 1960.

[9] J. Knight, A. Davison, and I. Reid. Towards constant time slam using postponement. *EEE/RSJ International Conference on Intelligent Robots and Systems*, pages 405–413, 2001.

[10] Kenneth Jay Lee. Reactive navigation for an outdoor autonomous vehicle. Tech. report, University of Sydney, 2001.

[11] Jean Lorieux. Detection et suivi de vehicules par telemetri laser en robotique autonome. Rapport de dea, Institut National Polytechnique Grenoble (INPG), INRIA Rhone Alpes (GRAVIR), Juin 2003.

[12] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem, 2002.

[13] Michael Montemerlo and Sebastian Thrun. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges.

[14] K. Murphy. Bayesian map learning in dynamic environments.

[15] J. Neira and J.D. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Trans. on Robotics and Automation*, 17(6):890–897, 2001.

[16] D. Schulz and W. Burgard. Probabilistic state estimation of dynamic objects with a moving mobile robot. *Robotics and Autonomous Systems*, 34(2-3), 2001.

[17] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous Robot Vehicles*. Springer, 1990.

[18] R.C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. Technical report tr 4760 , 7329, SRI, 1985.

[19] J. D. Tardos. Data association in slam. Technical report, Stockholm, KTH, 2002.

[20] J.D. Tardos, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 21(4):331–331, 2002.

[21] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[22] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[23] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998. also appeared in Autonomous Robots 5, 253–271 (joint issue).

[24] Sebastian Thrun. Exploration and model building in mobile robot domains. In *In Proceedings of the IEEE International Conference on Neural Networks*, 1993. IEEE Neural Network Council.

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

[26] L. Zhao and C. Thorpe. Qualitative and quantitative car tracking from a range image sequence. In *International Conference on Computer Vision and Pattern Recognition (CVPR'98)*, pages 496–501, June 1998.