Data Association for Simultaneous Localization and Mapping in Dynamic Environments using Multiple Model Approach

by

Rex H. Wong

A dissertation submitted to the Graduate Faculty in Electrical Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy at The City University of New York

2012

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy

Prof. Jizhong Xiao

Date | Chair of Examining Committee

Dr. Mumtaz Kassir

Date | Executive Officer

Prof. Zigang Zhu, Dept. of Computer Science

Prof. Yi Sun, Dept of Electrical Engineering

Prof. Michael Conner, Dept. of Electrical Engineering

Prof. Thao Nguyen, Dept. of Electrical Engineering

Supervisory Committee

The City University of New York

Abstract

Data Association for Simultaneous Localization and Mapping in Dynamic Environments using Multiple Model Approach

by

Rex H. Wong

Advisor: Prof. Jizhong Xiao

Simultaneous localization and mapping is the fundamental problem in mobile robotics. Data association is considered the most difficult part of this problem. The difficulties come from two major sources: the uncertainty of positions and measurements; and the dynamics of the environments. In addition, time-constraint and computational complexity further hinder the development of data association algorithms in real-time applications. In this work, we propose a feasible approach to handle the real-time data association problem in clutter and dynamic environment. The approach is three-folded. First, we characterize this problem as an optimization assignment problem with the constraint of one-to-one assignment in presence of clutter and interference. The mathematical models are based on Bayesian Joint Probability Data Association and using the linear programming algorithms to obtain the feasible assignment solutions. Second, we adopt the concept of multiple model system so that each tracking filter can adaptively estimate the target behavior more accurately when the mode of relative motion changes. Third, the incorrect decision on initialization may be revoked by a specific designed sliding window method and ranked assignment formulation. Because of its adaptability and cost-effectiveness, this approach can be applied for real-time SLAM applications. Simulation is performed to demonstrate the effectiveness of this method.

ACKNOWLEDGMENTS

I would like to thank all members of my doctoral committee, Prof. Xiao, Prof. Zhu, Prof. Sun, Prof. Nguyen, and Prof. Conner for their precious time and support for this study, especially to my doctoral advisor, Dr. Xiao, for his continuous support and guidance in the pursuit of this research. In addition, I like to extend my appreciation to all the fellow researchers and students in CCNY Robotic Laboratory, Dr. Flavio Cabrera-Mora, Dr. Ravi Kaushik, Dr. Clara Nieto-Wire, Mr. Samleo Joseph, and Mr. Xiaochen Zhang, for their tremendous help and encouragement. Besides, I also like to express my gratitude towards Dr. Yu Wang, Dr. Hossein Rahemi, and Dr. Shouling He, for their altruistic advice and support in my career.

I must express my heart-felt sentiment of gratefulness to my deceased parents who had given me a life and a healthy growth. Without them, I would not be able to make it this far. The similar gratitude also goes to my family members, especially to my sisters May and Grace, for their long term support and love during the hardship.

Finally I like to express my appreciation to CUNY Graduate Center for its convenient and advanced facility which provides a wonderful environment for research, and for its financial assistance via its Chancellor's Scholarship program during the years of my doctoral study.

Table of contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Preliminary Overview

Ever since the dawn of robots in human history, there are situations where the autonomous mobile agents are expected to replace the human operators in many dangerous missions, especially in hostile environments. This includes the missions such as search and rescue, surveillance, exploration, and some critical military operations. Nonetheless, knowing the environment is crucial to the success of those tasks. The knowledge about the environment may be translated into the geographic locations of the landmarks and the trajectory of the robot in terms of Cartesian coordinates presented in a global frame. So, in order to carry out many high-level tasks, a robot must be able to constantly update its own position and navigate to the destination according to its path plan. To achieve this, the robot needs a map for path planning. Unfortunately, in normal case, the map is either unavailable or obsolete. Then, the robot must depend on its onboard sensor to “see” and “map” the environment. However, the dilemma is, without knowing its location, a robot is unable to draw a global map. On the other hand, without a map, it is not able to localize itself. In other words, it is impossible to get an accurate map without knowing its own location; vice versa, it is impossible to localize itself without a reliable map. In view of this, the idea of simultaneous localization and mapping (SLAM) has been developed so that robots or autonomous vehicles can build up a map within an unknown environment or to update a map within a known environment (with a priori knowledge from a given map) while keeping track of their current location.

Simultaneous Localization and Mapping (SLAM) is the recursive process by using the robot pose estimate to determine the landmark locations, while at the same time, using these landmarks to improve the robot pose estimate. As the landmarks are repeatedly re-observed by the onboard sensor through a recursive process of prediction and updating, their locations become increasingly certain and a map gradually converges to become a consistent global map.

Due to the imperfection in robot odometer and floor condition, the transition from one position to another is most likely non-linear and perturbed by process noise. Likewise, the position sensing devices such as laser scanner or sonar are usually not perfect in sense of absolute accuracy. The information obtained from these onboard sensors is also usually corrupted by the noise which induces the uncertainty in observation. A normal way to treat this imperfection is to construct a stochastic model for these random variables, and take advantage of the covariance matrix which formulates the relationship between the variables within the framework of state space. By capitalizing on Bayesian conditional probabilistic filter theory and Markov assumption of complete state, all the state variables can be orchestrated into the framework of a stochastic model of SLAM, and this model is supposed to generate the posterior which can approximate the current state rather accurately.

Based on Bayesian stochastic model and Markov chains, the SLAM algorithm normally consists of two recursive phases. First phase is motion or process phase; the second phase is update or correction phase. Data association plays an important role between these two phases because the correct correspondence between the measurements from the robot sensors and the stored estimate landmark data (the basis for mapping) is

necessary for update phase. In fact, data association is the core procedures for all kind of feature-based SLAM algorithms in updating their state estimation.

Besides robotics, the data association is used in many other applications, such as air traffic control radar system, missile guidance, security surveillance, and medical imaging processing or pattern classification and recognition ...etc. However, there are some major difference in data association between robotic SLAM and other tracking applications. For example, for robotic SLAM, data association deals with static landmarks, the only motion should come from the robot itself; while for other tracking application, such as air traffic control or ground radar defense systems; data association deals with the fast-moving targets such as aircrafts, and the relative velocities are part of the motion model. Furthermore, the landmark locations of SLAM are estimated via the vehicle-landmark covariance matrix, therefore, they are indirectly interdependent. On the other hand, the flying targets tracked by ground radar are probabilistically independent from one another, and there is no correlation matrix between targets' positions. (For simplicity, target and landmark are used interchangeably in this work). Nevertheless, these applications have many things in common in terms of track initialization, maintenance, reduction, and termination. But, SLAM does not consider the problems in track forking, merging, and crossing that normally occur in air traffic control tracking problem.

Ideally, the data observed by the sensors should correlate with the landmark data currently stored in the robot's memory that forms a "map" as reference of robot's location. In clear environments the various features of the objects are discernable by their difference. However, in a cluttered or featureless environment, for example, office

building, museum, or factory, the similarity of interior structures makes distinguishing the features of landmark difficult. Besides, factors such as the imperfect sensing devices, uncertainties in vehicle pose, variable feature densities, objects moving around in the environment, or spurious sensor's signal reflecting from the walls, all of them can generate severe false alarm and interference that cause the anomaly in data association. The wrong data association is catastrophic for any application depending on tracking estimation, including SLAM.

Although the problem of data association has recently gained much attention within the SLAM research community, however, many proposed SLAM algorithms in the past still tended to assume that the data association between landmark features and measurements of observations is known. The troublesome unknown data association is either eluded or marginally investigated. The main reason is that data association is a NP-hard task in a dynamic scenario and finding a correct model for system estimation is very difficult in the context of SLAM [Thrun, S., Fox, D., and Burgard, W. 2005]. Some of proposed solutions for the problem are usually expensive in terms of complexity and cost from system point of view [Smith, Self, and Cheeseman.1990]. As the trend of decentralized robotics prevails in many applications, we see there is the urgent need for a simple and less expensive data association algorithm for a real-time robotic application. As such, we are motivated to take on our study with respect to this issue.

1.2 Objective of Thesis

In the context of SLAM, a successful data association should be able not only to correctly associate the observed measurement to its originating state estimate (i.e., landmark's position), to initialize the new landmarks, but also to prune out the false

measurements due to spurious signals such as clutter, or other interferences such as people or vehicles. Due to the complexity of this undertaking, these problems must be handled by different methodologies.

The objective is to make sure the correct data association is maintained between the static landmarks and the laser scanner's observations in spite of the interference made by these moving objects. This involves many issues. First, in static environment, due to the imperfection of sensing devices, the major problem of data association is the uncertainty regarding to the origin of the measurements [Sittler, 1964] and the uncertainty of the vehicle pose due to the imperfect sensors. In the crowded environment, if these landmarks are closely spaced, this imperfection would generate confusion. This situation is illustrated by the following scenario as shown in Figure 1.1, where a robot obtains two measurements from three adjacent landmarks. However, the measurement on the left side may come from either the landmark on the left or the landmark in the middle; while the measurement on the right side may come from either the central landmark or the landmark on the right side. These landmarks are competing with each other to associate with the possible measurement within their estimate boundaries. We can see this uncertainty is due to the noise not only from the sensor, but also from the robot's position. Besides, data association has to deal with spurious signals reflected from walls and the clutter that creates the false alarms to the sensor and thus degrade the data association.

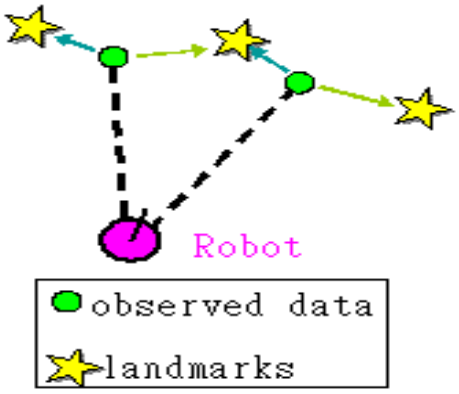


Figure 1.1: Data association problem corresponding confusion for the pair.

In addition, in case of the dynamic scenario, the additional challenge comes from the dynamic impact of the moving objects within the SLAM environment [Cox, 1991]. This dynamic characteristic has an adverse impact on the data association, especially on the new landmarks initialization. As a result, the ambiguous data association between the observation and landmark will result in the loss of the robot (not able to localize itself), or an inconsistent map (cannot close the loop if traversing circularly).

1.3 Contributions of Thesis

In this thesis, we introduce a novice methodology in constructing data association framework. It capitalizes on the concept of interactive multiple model (IMM) used in many tracking applications [Shalom et al. 1988] where the moving targets are the essential objects. But we only use it within certain time-depth, nominally, up to threes scans, for the purpose of differentiating the moving objects from the legal static landmarks. This way, we can avoid the entanglement of complicate tracking situation where the computational burden is extremely high and sometimes the solution is NP-hard due to the complexity. For each (landmark-centric) tracking filter, there is a bank (set) of model to adopt as the temporal estimator to process the data association and SLAM algorithm. The adoption of the model by the same filter may be different from time to time, depending on the dynamic situation. Within the model bank, each model is formed to describe the possible mode of dynamic motion between the robot and the observed objects. These motion models are constructed based on three factors:

1. Robot's kinematics which controls the motion model and the observation model.
2. Robot's own reactive motion due to the obstacle avoidance mechanism (OA).
3. Moving object's motion relative to the robot.

We formulate our tracking filters based on a Bayesian recursive framework as described in [Fortmann, 1983; Shalom; 1988; Neira, 2002], in which the probability of feasible events (correctly assigning an observation to the corresponding target or landmark) can be computed and form the basis for validating the data association. We modify the joint probability data association (JPDA) algorithm which takes the average of all measurements (with weights) within a validation region centered at the estimate location of the landmark by incorporating the probabilities of detection and false alarm from signal processing process (defined in the feature extraction algorithm) into the calculation of the measurement-landmark association probability. In doing so, the number of false association is reduced. For dynamic objects in the scenario, we adopt the concept of IMM (interactive multiple models) and carefully select the appropriate models to encompass the behavior mode of the moving objects so that we can better estimate state variables within the state space. We devise a novice "Clutter-like covariance sensitive detection driven mechanism of switching" to activate the model mode transition.

In case of abnormal situation, where the wrong decision on data association occurs and lead to huge error in SLAM, we use the idea of Q-best ranking assignment algorithm to rescind the decision of data association and replaced it with the next-best choices. We also devise an M out of N sliding window for new landmark initialization which is immune to the interference from the moving clutter.

In this study, we consider only the simplest infrastructure, that is, a single robot with only one sensor dedicated for SLAM process. Networked robotic scheme or multi-sensor data fusion is not studied here. However, it does not mean our methodology cannot be extended to the scenarios of multi-robots or multi-sensors applications.

1.4 Layout of Thesis

This thesis is organized as follows. In the following chapter, we review the relative research literatures regarding to the field of our study. We summarize the key issues and analyze their utility and importance in data association in different applications. In chapter 3, we present the preliminary groundwork for data association issue involved in the SLAM applications. In chapter 4, we describe the problem of interest and the formulation of mathematical models used for analysis. In chapter 5, we discuss the framework and key elements in the proposed data association algorithm and the scenario used to formulate our proposed NNJPDA, which incorporates a specific assignment optimization algorithm for feasible hypothesis events generation described in chapter 6. In chapter 7, we refine the algorithm by using Q-best 3S-2D assignment optimization to handle the more severe situation for landmark initialization and maintenance. To take our algorithm to the next level of complexity in the real dynamic environment, we resort to the multiple model concept used in many tracking applications. The performance evaluation and the results of simulation is discussed in chapter 9. We conclude our thesis in the final chapter and express our scope of interest in the future study.

Chapter 2

Background and Related work

In the past, there are many approaches in data association. At the low-end of the full spectrum of algorithms, it is the Nearest Neighbor (NN) filter. On the other hand, at the high-end of the spectrum, there is Multiple Hypotheses Tracking (MHT algorithm). NN is a classical technique [Fortmann, 1983; Shalom, 1988] in tracking problems. It associates the feature to the nearest observation in a chosen validation region based on some distance measure, called Mahalanobis distance [Sittler, 1964; Shalom, 1988]. The advantage of NN is its simplicity. It works well in sparse and clean environment. However, when operating in complex environments, where the uncertainty of landmark positions and the clutter level is high or landmarks are crowded in a tiny space, the NN algorithm may accept a wrong data association (False Alarm), which leads to the catastrophic failure in SLAM. Another algorithm similar to NN is called SN (Strongest Neighbor) which assumes the true measurement coming from the strongest signal amplitude. But SN only works for certain sensor with the radiated energy proportional with the distance. Similar to the NN filter, it also suffers from the ambiguity caused by noisy clutter in a congested environment as well.

MHT, on the other hand, is the most structured approach [Sittler, 1964; Shalom, 1988; Fortmann, 1983, Neira, 2002] which uses postponed decision logic [Cox and Leonard, 1991; Nagarajan, 1987, Reid, 1979].This logic scheme allows data association decisions to be delayed until additional scans of measurements are successively received so that incorrect associations made in the past can be revoked if necessary. In addition, MHT or its variants use different number of scans (sliding windows) to delay the decision

to initialize new tracks (or targets) if it is not clear in the first scan. For example, MHT defers the association decisions in conflicting situations and forms a tree of all possible association hypotheses, then propagated them through subsequent iterations in belief that new information will most likely resolve conflicts. In this case, MHT is capable of handling false alarms, missed detections, spurious signals and measurements, and new target initiations. Therefore, theoretically, MHT is an optimal solution to the multiple targets tracking in a cluttered and dynamic environment. However, the major drawback of MHT is that the hypothesis tree grows exponentially in time, and due to its high cost of computation and the complexity, implementation of MHT is impractical for a small robotic platform which normally is equipped with simple sensors and limited computational power and memory. In order to take advantage of MHT, but avoiding the computational burden, Davey [Davey, 2007] proposed an algorithm called "PMHT" (Probabilistic Multi-Hypothesis Tracker" which is claimed by the author to achieve the low computational complexity.

Between these two extremes, other algorithms have been developed. Many of them are based on PDA (probabilistic data association) and JPDA (Joint PDA). PDA [Shalom, 1988; Cox, 1991] and its variants use combined innovation from several possible measurements, but assuming only one is target-originated, and others are clutter. It can perform efficiently in a less dense environment, where the target validation regions do not overlap. But in case of dense environment, where the validation regions of the adjacent targets overlap, PDA fails to perform correctly due to the assumption that each target is independent and thus ignoring the fact that the SLAM model estimation is based on the correlation of landmarks, and their interdependence prevents any attempt to

decouple them into unrelated entities. On the other hand, JPDA also assumes that each target should produce at most one measurement, but it takes into account for the fact that any measurement which falls in the cluster of validation gates formed by several targets might have originated from any of these targets or from clutter within the scan area [Fortmann, 1983; Shalom, 1988]. Although the state estimation for each target is similar to PDA, the measurement-to-target association probabilities are computed jointly across the targets in JPDA. In this case, the validation gates are clustered by a common measurement which is located in the intersections of validation gates. The state vector estimate is updated with a pseudo-innovation that is the weighted average of the innovations for all sensor measurements falling within the validation region of the cluster in a given scan. The weighting for each measurement is computed from the conditional probabilities. The weights are based on the statistical likelihood of each measurement relative to the predicted landmark under consideration and other landmarks within the same cluster. The result shows a state estimate influenced by all neighboring landmarks and measurements within the clustered validation regions. In addition, the covariance of the state estimate accounts for the fact that not all the measurements contained within the validation gate of state estimate belong to the target of interest.

The typical JPDA incorporates probabilities of detection and false alarm in computation of association probability conditioned on the probability of feasible event hypothesis [Shalom & Li, 1995], and this feature makes it the more suitable choice for the cluttered situation. However, since the number of possible feasible hypotheses of data association between measurements and targets may grow exponentially as the number of the targets (landmarks and clutter together) increases, this optimal JPDA is too slow to

process the data set in real time. Besides, JPDA computes the probabilities of association only from the latest set of measurements. Therefore, they are not able to initialize the new targets from the clutter, nor can it distinguish the false alarm coming from the moving objects.

Many modifications have been proposed to reduce the computation time. Fitzgerald has proposed the "Cheap JPDA" [Fitzgerald, 1990] in which a simple ad hoc formula is developed to approximate the association probability to avoid the complexity and cost of computing the likelihood of all hypotheses like regular JPDA. However, this approximation sometimes causes confusion and results in incorrect solution when the number of measurements and tracks is equal. Besides, the association probability that CJPDA computes may not sum to unity and this can cause the target track to be drawn to a nearby target or false track. Roecker proposes a sub-optimal JPDA [Roecker, 1993], where the partial joint event calculation is suggested and results in a more accurate association probability than CJPDA. However, regardless of what version of JPDA, because of taking weighted average for all the measurements that fall within a target's validation region for target state update, track bias and coalescence occur when used in a dense target environment. In other words, the closely spaced targets tend to attract each other and merge into one [Fitzgerald, 1985]. In view of this problem, Fitzgerald again suggests a Nearest-Neighbor JPDA (NNJPDA) [Fitzgerald, 1985; 1990] which abandons the weight average updating of track in favor of one-one assignments of measurements to tracks. The association probabilities are used for making these one-to-one assignments. In NNJPDA, CJPDA has been used for calculating the association probabilities. In fact, any

JPDA technique can be used to compute association probability including sub-optimal JPDA.

The last concern about using JPDA is the need of an effective search algorithm for enumeration of all possible feasible hypotheses and solves it in order to get the best set of feasible hypotheses. If the targets in the cluster become excessive, total numbers of all feasible hypotheses will increase exponentially. The exhaustive search for feasible hypotheses is NP-hard and unable to provide solution in real-time [Nagarajan, 1987]. Instead of enumeration of all feasible hypotheses, the sub-optimal algorithms have been developed. Zhou et al. proposed a Fast-JPDA [Zhou, 1993] which is based on Depth First Search (DFS) algorithm, using measurement-oriented concept. DFS adopts a back-track method to reduce the search for the feasible hypotheses without exhaustive enumeration. Wijesoma et al. [Wijesoma, 2004] presents a MDA (Multi-Dimensional Assignment Data Association) method which formulates the SLAM data association as a non-linear discrete optimization problem in two consecutive scans to determine new tracks and exclude moving objects. However, the joint likelihood function in this method involves too many partitions of hypothesis space and the calculation may again grow without bound in a high false alarm environment. Besides, the improper target deletion may occur if the tentative target which has been detected in previous time frame and not seen in the current scan (time frame). All in all, regardless of the above effort, the combinatorial nature [Nagarajan, 1987; Blom, 1988] of this problem requires a more systematical and robust approach to tackle.

In view of this ordeal, Murty [Murty, 1968] proposed the ranked assignment optimization. Miller [Miller, 2001] and Popp [Popp, 2001] followed the same philosophy

to propose a K-best ranked S-D assignment algorithm for data association in multi-target tracking problem that guarantees the complete assignment between measurements and targets within a polynomial time, regardless of whether the assignment set is optimal (minimum cost) or suboptimal (with slack relaxation on the minimal cost constraint). However, association of the moving objects or semi-moving objects (such as constantly open or closed doors) to real targets (static landmarks) still remains an unsolved issue if the estimation model cannot accommodate all possible motions of the tracked targets.

The interactive multiple model (IMM) estimator has been proved to be more effective in handling the tracking problem of dynamic scenario [Shalom and Li, 1995; Li, 2005]. There are two major categories of IMM design. One is the fixed model system; another is variable model system. The fixed model system assumes that the target behavior is defined by the mode matched to one of a finite set of predetermined models, and the model in use is assumed to evolve through the modes according to a Markov chain [Blom, 1998]. By having probabilistically combined estimates of the individual filters matched to these modes, the overall estimate can be obtained. The state vector and the covariance of the filter are pre-determined before the estimation is carried out. Since the dynamic motion in the complex environment is not predictable and the system requires a large number of models to fully encompass all possible behaviors (i.e., the maneuver motions of a target), this kind of design becomes infeasible in real time application. The more appropriate design is the variable IMM which can adapt to the variation of target modes and thus requires less number of models to track the targets [Shalom, Li, and Kirubarajan, 2001;]. Since the initialization and termination are two important issues and they can not be solved by the algorithm which takes account for the

measurements only from current scan. For this matter, the sliding window with a proper time-frame depth used in a multi-scan IMM can not only accurately estimate the data association in the cluttered and confused situation, but also help correctly initialize the new target or landmark, while pruning out the redundant and obsolete ones [Wijesoma, 2004; Li, 2005].

In this paper, we propose a data association scheme for SLAM. The primary framework is built based on the concept of IMM, which encompass the mathematical models sufficient to describe the all possible modes of target motion with pertaining process noise. It also formulates the mode switching mechanism in terms of the model-in-effect probability, the transition probability, and other parameters. For the measurement model and the algorithm for data association, we adopt the idea of NNJPDA in order to reduce the computational cost and to quicken the filtering process for the real-time application. As for feasible hypothesis generation, we follow the philosophy of Murty's ranked assignment for optimization problem to formulate our Q-best 3S-2D assignment problem relaxation method to solve problem in polynomial time. The benefit of using IMM with this type of assignment and NNJPDA can be manifested by comparing it with other algorithms in terms of complexities, RMS position errors, loss of tracks, and time for convergence.

Chapter 3

Preliminary

3.1 Simultaneous localization and mapping (SLAM)

An important feature for a mobile robot of SLAM capability is the ability to build and maintain maps of initially unknown environments. Maps allow the vehicle to plan its movement within the environment in order to achieve its goals. The automatic creation of maps allows the vehicle to model the world using its sensory resources. In case of SLAM, the creation of an accurate map of the environment is a goal in itself while in many other circumstances the maps are used as a tool for performing other higher level tasks.

Given accurate positioning information, the creation of a map of the environment is a straightforward undertaking. Given a current position estimate and an observation of the environment the observation can be fused into the existing map to create an updated map estimate. Many grid-based approaches have relied on this type of map-building in order to generate an occupancy map of the environment [Siegwart and Nourbakhsh, 2004]. A number of other mapping techniques rely on the extraction of relevant environmental features with which to build maps [Thrun, Fox, and Burgard, 2005; Castellanos et al, 1999]. Most feature maps consist of a set of landmarks and encode some relationship between the features contained in the maps. The relationships are often geometrical in nature but, given appropriate sensors, might also encode other properties of the landmarks such as color, texture or shape, as in vision-based sensing system.

Before we commence the development of our SLAM/DA algorithm, a thorough analysis of the scenario of mapping and environmental conditions should be done so that it may provide great help in determining how to apply filter(s) in the stochastic model.

3.2 Maps and Scenario

Environment can be represented by two types of maps. First one is called metric map, which contains the occupancy grids (Fig.3.1a), and the feature of landmarks (fig.3.1b). The other is called topological map (Fig.3.1c). The former is of geographic nature which is based on the metric grid cell and the "feature" of environmental landmarks whose locations are represented by a set of Cartesian coordinates or some specific markings. As for the topological map, it does not rely on metric measurements and instead represents the environment in terms of nodes and the paths connecting them. They are generally depicted by a graph where the nodes define particular locations in the environment and the graph edges define procedural information for traveling between nodes.

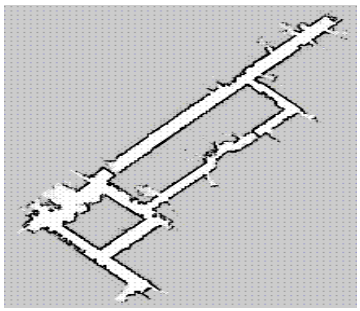
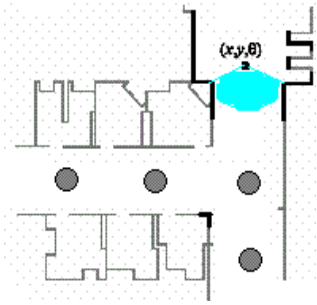
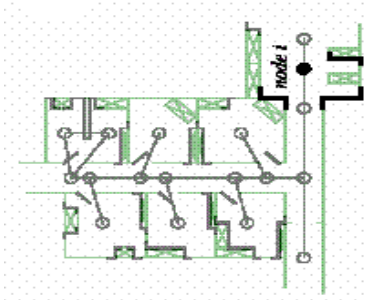


Figure 3-1(a) Occupancy grid map



(b) Feature map



(c) Topological map

For localization, a stochastic model can initialize the positions of an occupancy grid and update the estimate posterior by incorporating the "features" of landmarks. For mapping, the "local" maps obtained from sensor observations can be fused into a "global" map to complete the process. The local map refers to the robot's local frame, while the global map refers to the world frame of coordinate.

3.2.1 Occupancy grid maps

The occupancy grids of the metric map represent a region as a matrix of cells. Each cell

is a small rectangular area in the environment, and provides an explicit representation of both occupied and free space, which is useful for path planning. Although occupancy grids are reasonably able to incorporate models of sensor uncertainty and so are suitable for either localization given an *a priori* map or map building given location. However, SLAM requires an integrated representation of sensor and vehicle pose uncertainty and their correlations, and this is not supported within the occupancy grid framework because occupancy grids can represent uncertainty at a local (vehicle-centric)level, but not at a global level—which is essential for map convergence. By not defining criteria for convergence, the developing map is easy to drift with each observation update and this divergence exhibits itself as a slow blurring of the map. This condition gets worse when it comes to deal with the cyclic trajectory. This is why occupancy grid is not suitable for mapping a large scale environment.

3.2.2 Feature map

On the other hand, the feature maps (or landmark maps) represent the environment by the global locations of geometric features such as points and lines. Localization is performed by extracting features from sensed data and associating them to features in the map. The differences between the predicted feature locations and the measured locations are then used to calculate the vehicle pose. In this way, localization is pretty much like a multiple target tracking problem, but, unlike tracking of moving targets, the targets are static and the observer is in motion. The landmark locations in an *a priori* feature map are assumed to be perfectly known and so each feature is entirely defined by its parameter set. For example, a point-location feature for a cylindrical landmark, such as a pole or tree trunk, might be defined by $\mathbf{f} = (c, r, x, y)$, where c is the feature type (cylinder type), r is

the cylinder radius, and x and y define its centre location. Only the location information is directly useful for localization but the other information serves to assist landmark recognition for data association. Since each landmark is represented by a limited set of parameters, the feature map $\{\mathbf{f}_1 \ldots \mathbf{f}_n\}$ is a very efficient environment representation. Unlike occupancy grids, where a dense environmental description is maintained, feature maps form a sparse representation of select landmarks. However, the free-space is not represented; therefore feature maps alone do not facilitate tasks like path-planning or obstacle avoidance that must be performed as separate operations.

Feature map SLAM comprises the dual task of adding observed features to the map, using the vehicle pose as a reference, while using existing map features to estimate the vehicle pose. The uncertainty of sensor measurements, therefore, results in uncertain estimates of both the vehicle pose and the map feature locations, and these uncertainties are dependent (or correlated), as shown in Figure 3.2. Correlated uncertainty has an important consequence for feature-based SLAM as it couples the individual features to each other and the vehicle to the map. On the other hand, attempts to estimate the vehicle pose and map features independently have been shown to produce inconsistent (optimistic) uncertainty estimates.

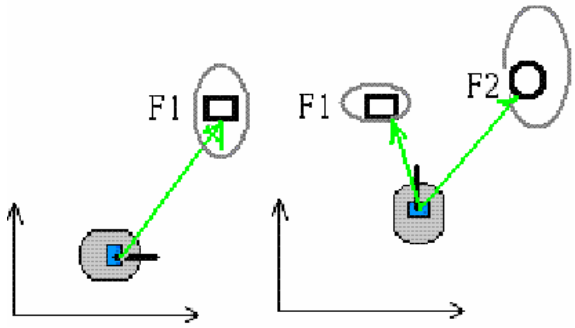


Figure 3-2 SLAM uncertainty correlations between vehicle and features

Since feature map SLAM is a parameter estimation problem which determine the vehicle pose (x, y, φ) given the map feature information and a set of feature observations.

Assuming the measurements are correctly associated to the appropriate map features, the vehicle pose can be tracked using standard estimation techniques. Consistent stochastic estimation requires that correlations between parameters are explicitly maintained. For an stochastic tracking filter, this means storing the parameters in a single state vector and their associated correlations in a covariance matrix. For feature-based SLAM, the vehicle pose estimate and the map feature locations must be stored in the same state vector, and this vector must be augmented as new features are observed and added to the map. Assuming the SLAM process satisfies the basic conditions of near-linearity and approximately Gaussian uncertainty distributions, and then the uncertainty model provided by the stochastic tracking filter can yield monotonic map convergence. With repeated observation and update, the map becomes certain such that the relative feature locations are perfectly known and the global uncertainty is reduced to a lower bound.

However, correct pose estimation relies on finding correct correspondence between a feature observation and its associated map feature. A wrong association results in an inconsistency, where the vehicle pose uncertainty decreases, but the estimate error actually increases. The effect of significant false associations is to dramatically increase the pose estimate error and prevent any subsequent map registration so that the vehicle becomes lost. Most feature map localization implementations are susceptible to data association failure because they rely on the association methods developed for target tracking.

Furthermore, feature maps is suitable only to the environment where the observed objects can be presumably depicted as basic geometric feature models. This is often not the case in an unstructured environment where the observed objects might appear to be any arbitrary curves rather than distinct points or lines. For reliable operation in these environments, it is necessary to devise parametric feature models that describe these general objects with fidelity for consistent extraction and classification.

After all, feature map is still a viable representation for stochastic SLAM in fairly small-scale environments where few stable landmarks are observable, computation is tractable, and accumulated state uncertainty does not exceed conservative limits. For the operation in larger areas, modifications to the basic stochastic SLAM algorithm are required.

3.2.3 Topological map

The topological map is a major conceptual shift in representing the environment. Occupancy grids and feature maps are both metric maps where location is defined as a set of coordinates in Cartesian space. Topological maps do not rely on metric measurement, and they are generally represented by a graph structure where the graph nodes define particular locations in the environment and the graph edges define procedural information for traveling between nodes. Thus, navigation between two non-adjacent locations is determined by a sequence of transitions between intermediate place nodes. For example, in Figure 3.3, the environment is defined using a graph data structure where each node (or vertex) contains a place description and each edge contains a path description. Traveling from one place to another entails traveling via intermediate places and standard

graph shortest path algorithms can be used (e.g., A to H requires traveling through the sequence A-C-D-F-H or A-B-E-F-H).

Topological maps, as an *a priori* reference, are attractive because of their efficient and compact representation, and therefore it is good for tasks like path planning.

Topological map is irrelevant to the pose uncertainty estimation, and uses only the qualitative measures for navigation like "follow path from A to B" or "at B". It has the advantage by using standard graph algorithms for high-level planning operations such as finding the shortest path between non-adjacent nodes [Castellanos, 1959].

The primary weakness of topological maps concerns ensuring reliable navigation between places, and subsequent place recognition, without the aid of some form of metric

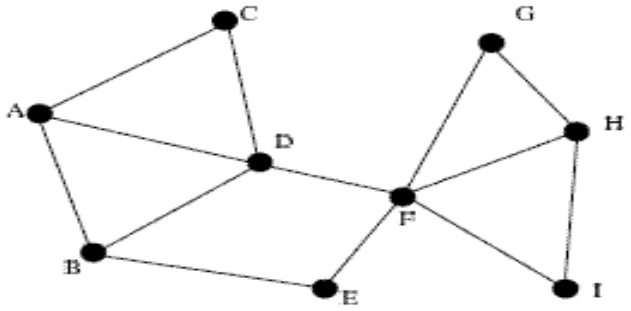


Figure 3-3: the topological graph with nodes and edges

location measure. Traveling between nodes using pure trajectory information, such as way-points, is often sufficient for static structured environments. But in more complex and dynamic environments, it may fail to guide the robot to the destination. The most critical weakness, however, is place recognition. If a place is not recognized (missing detection) or an alternate location is mistaken for a place (false alarm detection) then the topological sequence is broken and the robot becomes lost. Missing detection occurs because of alteration of appearance through circumstances such as viewpoint variation,

occlusion, structural change, dynamic objects or changed lighting conditions. Both geometric and visual recognition methods are sensitive to this form of failure. False alarm detection can be generated if another portion of the environment appears similar to the place definition—meaning that the place is not locally unique. This is common in highly structured environments (such as rows of office cubicles) but can also be a symptom of inadequate place definition. For the simple descriptions offered by most geometric recognition methods, ambiguous associations may be created even by the presence of transient objects. Vision-based recognition is probably more immune to false alarm because of the increased level of information defining the node.

By avoiding metric location measurement, topological SLAM removes the difficulties of uncertainty representation and problem of non-linearity. However, it places full weight for robust operation upon data association. In the case of cycle detection, where an observed place is found to resemble a previous place (or perhaps several previous places), data association becomes ambiguous, and the observed place could be one of the stored locations or a newly discovered location.

Another problem with topological maps is their limitation to waypoint-based navigation. The robot is constrained to follow specific trajectories and to pass through (or near) each node location. Control is directly tied to the localization process (i.e., the robot must perform active localization). While this situation is adequate for many autonomous vehicle systems, there exist applications where the robot trajectory should be independent of discrete place locations and passive localization is necessary.

3.2.4 Hybrid maps

Hybrid maps would take the advantages of both metric and topological maps since the

qualities of metric and topological maps are complementary. Metric maps, with an appropriate uncertainty representation, constrain data association and permit non-qualitative trajectory planning, while topological maps break the world up into locally connected regions and avoid the problems of maintaining a global reference frame. Hybrid topological-metric maps [Williams, 2001; Simhon,1998] are basically topological frameworks where the place definitions and/or path definitions contain metric (as well as qualitative) information. Importantly, this means that places are no longer restricted to discrete locations but can describe regions of arbitrary size and shape as local metric maps. By adopting hybrid topological-metric structure, the problems of computational cost and non-linearity can be alleviated by decoupling distant (non-adjacent) regions and treating only the sub-map locally. Cycle detection remains a significant problem but, by combining place recognition methods with pose uncertainty constraints, it is possible to resolve the loop-closure problem in cyclic scenario.

3.2.5 Stochastic map and state vectors

A stochastic map contains estimates of the spatial relationships, their uncertainties, and their inter-dependencies [Smith, Self, and Cheeseman, 1987]. A spatial relationship will be represented by the vector of its spatial variables, i.e., $\mathbf{x}$ =[x, y, θ]', its expected value $\hat{x}$ and its covariance Σ. The locations of 2-D point features observed by the vehicle form a map in the same base coordinate system (More elaborate parametric feature models, such as lines, might also be used). The covariance matrix $\mathbf{\Sigma}_m$ of this map includes cross-correlation information between the features (i.e., the off-diagonal terms) that interweave the dependence of each feature location given the knowledge of the other features in the

map. Since the feature locations are static, these correlations will increase when re-observed by the sensor and the map becomes increasingly converged and certain.

$$\mathbf{x}_{\mathrm{m}} = \begin{bmatrix} x_1 & y_1 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}; \mathbf{\Sigma}_{\mathrm{m}} = \begin{bmatrix} \sigma_{x1x1}{}^2 \sigma_{x1y1}{}^2 & \cdots & \sigma_{x1xn}{}^2 \sigma_{x1yn}{}^2 \\ \sigma_{x1y1}{}^2 \sigma_{y1y1}{}^2 & & \sigma_{y1xn}{}^2 \sigma_{y1yn}{}^2 \\ & \ddots & \\ \sigma_{x1xn}{}^2 \sigma_{y1xn}{}^2 & & \sigma_{xnxn}{}^2 \sigma_{xnyn}{}^2 \\ \sigma_{x1yn}{}^2 \sigma_{y1yn}{}^2 & \cdots & \sigma_{xnyn}{}^2 \sigma_{ynyn}{}^2 \end{bmatrix} \tag{3.1}$$

Therefore a stochastic map can be defined by an augmented state vector formed by the concatenating the vehicle state and the map feature state. The initial state vector and its covariance is represented by

$$\hat{x}_0 = \begin{bmatrix} \hat{x}_v \\ \hat{x}_m \end{bmatrix}; \ \Sigma_0 = \begin{bmatrix} \Sigma_{\mathrm{v}} & \Sigma_{\mathrm{vm}} \\ \Sigma_{\mathrm{vm}} & \Sigma_{\mathrm{m}} \end{bmatrix} \tag{3.2}$$

(Note that the initial condition of the state estimate is usually given as $\hat{x}_0 = \hat{x}_v = \mathbf{0}$ and $\mathbf{\Sigma}_0 = \mathbf{\Sigma}_v = \mathbf{0}$. In other words, no features have yet been observed and the initial vehicle pose defines the base coordinate origin.)

3.2.6 The correlation matrix of SLAM

The most important insight in SLAM was to manifest that the correlations between landmark estimates increase *monotonically* as more and more observations are made. This means that knowledge of the relative location of landmarks always improves and never diverges, regardless of robot motion. Referring to Figure 3.4, it can be seen that the error between estimated and true landmark locations is due to the fact that a single errors in knowledge of where the robot is when landmark observations are made. In turn, this implies that the errors in landmark location estimates are highly correlated. Furthermore the relative location between any two landmarks, say, $\mathbf{m}_i$, $\mathbf{m}_j$, may be known with high accuracy, even when the absolute location of a landmark $\mathbf{m}_i$ is uncertain. Put in a

probabilistic form, this means that the joint probability density for the pair of landmarks $P(\mathbf{m}_i,\mathbf{m}_j)$ is high even when the marginal densities $P(\mathbf{m}_i)$ may be quite dispersed. Consider the robot at location $\mathbf{x}_k$ observing the two landmarks $\mathbf{m}_i$ and $\mathbf{m}_j$. The relative location of observed landmarks is independent of the coordinate frame of the vehicle, and successive observations from this fixed location would yield further independent measurements of the relative relationship between landmarks. As the robot moves to location $\mathbf{x}_{k+1}$, it again observes landmark $\mathbf{m}_j$ this allows the estimated location of the robot and landmark to be updated relative to the previous location $\mathbf{x}_k$. In turn, this propagates back to update landmark $\mathbf{m}_i$—even though this landmark is not seen from the new location. This occurs because the two landmarks are highly correlated (their relative location is well known) from previous measurements. Further, the fact that the same measurement data is used to update these two landmarks makes them more correlated. At location $\mathbf{x}_{k+1}$, the robot observes two new landmarks *relative* to $\mathbf{m}_j$. These new landmarks are thus immediately linked or correlated to the rest of the map. Later updates to these landmarks will also update landmark $\mathbf{m}_j$ and through this landmark $\mathbf{m}_i$ and so on. Finally, all landmarks end up forming a web linked by relative location or correlations whose precision or value increases whenever an observation is made.

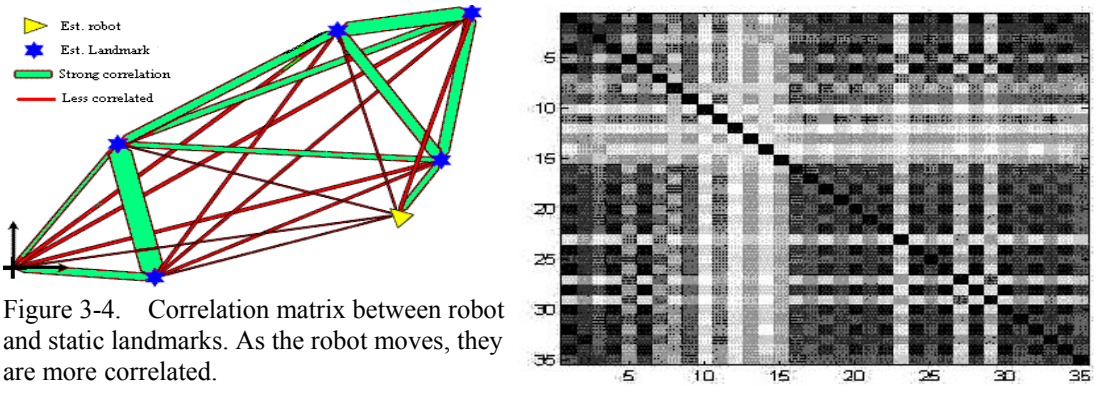


Figure 3-4. Correlation matrix between robot and static landmarks. As the robot moves, they are more correlated.

Chapter 4

Problem Formulation and System Modeling

The data association for indoor feature-based SLAM faces many challenges. The first one is the clutter problem which arises from scattering spurious signals reflected from the walls or objects within the environment. The second problem is the densely spaced geometric features that cause confusion in identifying landmarks. The third challenge comes from the dynamic objects temporally moving in and out of the scene during data acquisition, and this challenge applies to both indoor and outdoor scenarios.

In this chapter, we analyze the layers of data association problems regarding to SLAM and formulate the groundwork for mathematic model to tackle the problems.

4.1 Probabilistic SLAM model and Recursive Bayesian Estimation

In the framework of SLAM, the raw data set is processed by the feature extraction or the signal processing algorithm of the measurement device(s) such as laser scanner, and then output to the data processing unit for further decision making and state estimation processing. For any 2-D scenario, the processed data set is a collection of geometric data

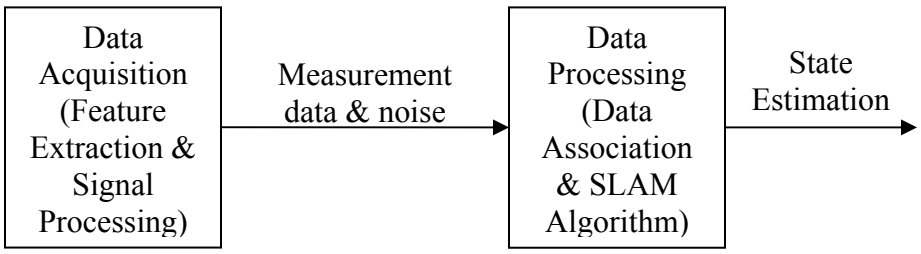


Figure 4.1 System block diagram of data association and SLAM.

of scanned landmarks. This procedure is illustrated in Figure 4.1.

Since data association is part of data processing where the decision is made based on the probabilistic data matching criteria, therefore it is usually considered as the state

estimation problem. It takes the data output from the feature extraction filter (signal processing) as measurements and associates them with the proper landmark features according to certain rules. Once the assignment is made, the SLAM filter can process the update of state variables.

The most common way to formulate the stochastic SLAM is by using a recursive Bayesian estimation model [Thrun, Fox, and Burgard, 2005]. It is implemented by storing the vehicle pose and map landmarks in a single state vector, and estimating the state parameters via a recursive process of prediction and observation. The prediction stage deals with vehicle motion based on incremental dead reckoning estimates, and increases the uncertainty of the vehicle pose estimate. The observation, or update, stage occurs with the re-observation of stored features, and improves the overall state estimate. When a feature is observed for the first time, however, it is added to the state vector through an initialization process. The Bayesian SLAM model can be represented by

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_o) \sim P(\mathbf{z}_k|\mathbf{x}_{k-1}, \mathbf{m}) \times P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_o) \quad (4\text{-}1)$$

that describes the joint posterior density of the landmark locations, $\mathbf{m}$, and vehicle state, $\mathbf{x}$ (at time k) given the recorded observations, $\mathbf{Z}_{0:k}$, and control inputs, $\mathbf{U}_{0:k}$, from initial

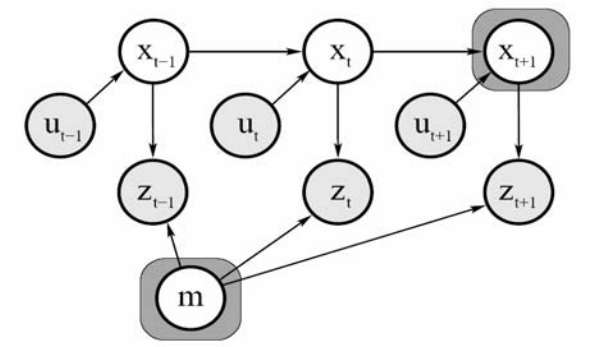


Figure 4.2. The Bayesian Model of SLAM

time up to and including time k together with the initial state of the vehicle, $\mathbf{x}_0$. This computation requires a state transition model and an observation model describing the effect of the control input and observation respectively.

The state transition is assumed to be a Markov process in which the next state $\mathbf{x}_k$ depends only on the immediately preceding state $\mathbf{x}_{k-1}$ and the applied control $\mathbf{u}_k$ and is independent of both the observations and the map. The observation model describes the probability of obtaining the sensor measurements $\mathbf{z}_k$ based on the vehicle position $\mathbf{x}_k$ and previous known landmark locations $\mathbf{m}$. The stochastic SLAM model is usually expressed as the posterior probability distributions in a two-stage recursive process:

1. Prediction Stage (motion or transition model)

$$P(\mathbf{x}_k,\mathbf{m}|\mathbf{Z}_{0:k-1},\mathbf{U}_{0:k}, \mathbf{x}_0) = \int P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \times p(\mathbf{x}_{k-1},\mathbf{m}|\mathbf{Z}_{0:k-1},\mathbf{U}_{0:k-1}, \mathbf{x}_0)d\mathbf{x}_{k-1} \quad (4\text{-}2)$$

2. Correction Stage (measurement update model)

$$P(\mathbf{x}_k,\mathbf{m}|Z_{0:k},\mathbf{U}_{0:k}, \mathbf{x}_0) = P(\mathbf{z}_k|\mathbf{x}_k , \mathbf{m})P(\mathbf{x}_k ,\mathbf{m}|\mathbf{Z}_{0:k-1}, U_{0:k} , \mathbf{x}_0)/P(\mathbf{z}_k |\mathbf{Z}_{0:k-1}, U_{0:k}) \quad (4\text{-}3)$$

Equations (4-2) and (4-3) provide a recursive procedure for calculating the joint posterior $P(\mathbf{x}_k,\mathbf{m}|\mathbf{Z}_{0:k-1},\mathbf{U}_{0:k}, \mathbf{x}_o)$ for the robot state $\mathbf{x}_k$ and map $\mathbf{m}$ at a time k based on all observations $\mathbf{Z}_{0:k}$ and all control inputs $\mathbf{U}_{0:k}$ up to and including the current time k. The recursion is a function of the vehicle model, $P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k)$, and the observation model, $P(\mathbf{z}_k| \mathbf{x}_k, \mathbf{m})$. As it is well known, the Bayesian filter is a NP-complete recursive process and difficult to be fully implemented in real-time. In order to get the feasible solution, we have to approximate (4-2) and (4-3) by certain modeling algorithms.

4.2 The filters to approximate the stochastic SLAM problem

In order to implement the probabilistic SLAM problem, we need to find an appropriate mathematical form for both the motion model and observation model that allows efficient

and consistent computation of the prior and posterior distributions (Eq. 4-2 and 4-3). To date, the most common representation is a state-space model with additive Gaussian noise. The Kalman filter and its non-linear derivative, Extended Kalman filter (EKF) is the most commonly used baseline model for implementation of probabilistic SLAM. To alleviate the linearization burden, the Unscented Kalman filter (UKF) scheme is proposed. For the highly non-linear scenario, a general non-Gaussian particle filter is used. There are many variants of particle filter, such as Rao-Blackwellized particle filter [Doucet, 1998], and FastSLAM [Montemerlo, 2002]. The latter becomes very popular in SLAM research community and has drawn more and more attention recently. It uses particle filter for robot's localization and EKF for estimating landmarks' position. Therefore it is worthwhile to investigate the EKF-SLAM model and its data association.

4.2.1. EKF-SLAM: a Baseline SLAM model

As shown in Figure 4-3, the EKF-SLAM can be modeled by the following scenario. A vehicle is moving relative to its previous pose according to the encoder of odometer's motion estimate. The vehicle motion is modeled as

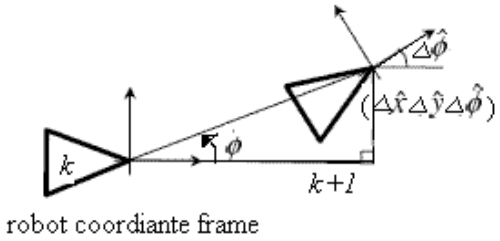


Figure 4-3 *SLAM Motion model*

$$P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \Leftrightarrow \mathbf{x}_k = \mathbf{g}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \boldsymbol{\varepsilon}_k, \tag{4.4}$$

where $\mathbf{g}(\cdot)$ models vehicle kinematics and where $\boldsymbol{\varepsilon}_k$ are additive, zero mean uncorrelated Gaussian motion disturbances (a white noise) with covariance Σ_k. The observation model is described in the form as

$$P(\mathbf{z}_k|\mathbf{x}_k,\mathbf{m}) \Leftrightarrow \mathbf{z}_k = \mathbf{h}(\mathbf{x}_k,\mathbf{m}) + \boldsymbol{\delta}_k, \tag{4.5}$$

where $\mathbf{h}(\cdot)$ describes the geometry of the observation and where $\boldsymbol{\delta}_k$ are additive, zero mean uncorrelated Gaussian observation errors with covariance $\mathbf{Q}_k$. With these definitions, the mean and covariance in Eq. (4-4) are, respectively

$$\hat{x}_k = \begin{bmatrix} \hat{x}_v \\ \hat{x}_m \end{bmatrix}; \quad \Sigma_k = \begin{bmatrix} \Sigma_v & \Sigma_{vm} \\ \Sigma_{vm} & \Sigma_m \end{bmatrix} \tag{4-6}$$

of the joint posterior distribution $P(\mathbf{x}_k,\mathbf{m}|\mathbf{Z}_{0:k},\mathbf{U}_{0:k}, \mathbf{x}_0)$ can be computed. Then, for the predicted augmented state can be described as

$$\hat{x}_{k|k-1} = g(\hat{x}_{k-1|k-1}, u_k) = \begin{bmatrix} \hat{x}_{v,k-1} + \Delta x \cos\hat{\phi} - \Delta y \sin\hat{\phi} \\ \hat{y}_{v,k-1} + \Delta x \sin\hat{\phi} + \Delta y \cos\hat{\phi} \\ \hat{\phi} + \Delta\hat{\phi} \\ \hat{\mathbf{x}}_m \end{bmatrix} \tag{4-7}$$

$$\Sigma_{v,k|k-1} = \nabla g_v \, \Sigma_{v,k-1|k-1} \, \nabla g_v{}^T + \Sigma_{k-1} \tag{4-8}$$

Where ∇g_v is the Jacobian of transformation g_v evaluated at the estimate $\hat{x}_{k-1|k-1}$. Since the landmarks are stationary, they are time-invariant, and therefore no time-update is needed. However, it is necessary to update for dynamic landmarks that may move.

For features of landmark, this is observed by the sensor with the measurements,

$$z_k = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \quad \text{, with the covariance } Q_k = \begin{bmatrix} \sigma_{rr}^2 & \sigma_{r\theta}^2 \\ \sigma_{r\theta}^2 & \sigma_{\theta\theta}^2 \end{bmatrix} \tag{4-9}$$

where r, θ are range and bearing relative to the observing sensor and Q_k is the observation covariance. Then, we can relate the observed information to the map as

$$\hat{z}_i = h(\hat{x}_{k|k-1}) + \delta_k = \begin{bmatrix} \sqrt{(x_i - \hat{x}_v)^2 + y_i - \hat{y}_v)^2} \\ \arctan(\frac{y_i - \hat{y}_v}{x_i - \hat{x}_v}) - \hat{\phi}_v \end{bmatrix} + \boldsymbol{\delta}_k \tag{4-10}$$

Assuming data association of the observation **z** to the map feature estimate $(\hat{x}_i, \hat{y}_i)$ is correct, the Kalman gain **K***i* can be derived by innovation vector and its covariance matrix as follows:

$$v_i = z - h_i(\hat{x}_{k|k-1}) \tag{4-11}$$

$$S_i = \nabla h_i(\hat{x}_{k|k-1}) \Sigma_v \nabla h_i(\hat{x}_{k|k-1})^T + Q_k \tag{4-12}$$

$$K_i = \Sigma_v \nabla h_i(\hat{x}_{k|k-1})^T S_i^{-1} \tag{4-13}$$

Where the Jacobian $\nabla h_i(\hat{x}_{k|k-1})$ is given by

$$\nabla h_i(\hat{x}_{k|k-1}) = \frac{\partial h_i}{\partial x_v}\bigg|_{\hat{x}_{k|k-1}} = \begin{bmatrix} -\Delta x/d & -\Delta y/d & 0 & \cdots & \Delta x/d & -\Delta y/d & 0 & 0 \\ \Delta y/d^2 & -\Delta x/d^2 & -1 & & -\Delta y/d^2 & \Delta x/d & 0 & 0 \end{bmatrix} \tag{4-14}$$

where $\Delta x = \hat{x}_i - \hat{x}_v$; $\Delta y = \hat{y}_i - \hat{y}_v$; $d = \sqrt{(\hat{x}_i - \hat{x}_v)^2 + (\hat{y}_i - \hat{y}_v)^2}$. Therefore, *a posterior* SLAM estimate is subsequently determined from the above update equations as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_i \times v_i \tag{4-15}$$

$$\Sigma_{k|k} = \Sigma_{k|k-1} - K_i S_i K_i^T \tag{4-16}$$

The observation model in Equation (4-11) correlates the feature estimate to the vehicle pose estimate and serves to reduce the uncertainty of both. Through correlation to the vehicle pose estimate, the map features become correlated to each other and these correlations increase monotonically until their locations (relative to each other) become perfectly known.

4.2.2 Alternative model: UKF-SLAM

The problem for EKF lies in its linearization process for non-linear state transformation. By using only first order coefficients of Taylor series evaluated at the nominal points, the EKF may not be able to fully approximate the highly non-linear system model, and therefore leads to some degree of inconsistency in map convergence. Another weakness of EKF is the computational burden of calculating the Jacobian matrix.

The unscented Kalman filter (UKF) [Simon and Uhlmann, 1997] uses a deterministic sampling technique known as the unscented transform (UT) to pick a minimal set of sample points (called sigma points) around the mean. These sigma points are then propagated through the non-linear functions and the covariance of the estimate is then recovered. UT deterministically extracts these sigma-points from the Gaussian distribution and propagates them through a non-linear function, g(x). These points are located at the mean and symmetrically along the main axes of the covariance.

The result is a filter which more accurately captures the true mean and covariance. (This can be verified using Monte Carlo sampling or through a Taylor series expansion of the posterior statistics.). In addition, this technique removes the requirement to analytically calculate Jacobians, which for complex functions can be a difficult task in itself. The benefit of UKF extends to systems of these natures: multi-modality, asymmetry, and discontinuity. The formulation of the UKF model is briefly described in Appendix A. (Detailed derivation of UKF can be found in reference [Uhlmann, 1997] and [Thrun, Fox, and Burgard, 2005]).

The filter parameters for comparison between EKF and UKF is listed in Table 4-1 below:

Table 4-1. The comparison of parameters of measurement model between EKF and UKF

Filter	EKF	UKF
Innovation vector: ***v***	$z - h_i(\hat{x}_{k\|k-1})$	$\gamma^i - \hat{z}_k$
innovation covariance: ***S***	$\nabla h_i(\hat{x}_{k\|k-1})\Sigma_v \nabla h_i(\hat{x}_{k\|k-1})^T + Q_k$	$\sum W_c^i(\gamma^i - \hat{z}_k)(\gamma^i - \hat{z}_k)^T$
Kalman Gain: ***K***	$\Sigma_v \nabla h_i(\hat{x}_{k\|k-1})^T S_i^{-1}$	$\Sigma_{x,z} S_k^{-1}$

It is shown that by using only the UKF, we can obtain the necessary parameters without going through the tedious and error-prone Jacobian of EKF (represented by the symbol ∇), although the computation cost may be the same. In the following chapters, the importance of these parameters will be manifested when calculating the validation gating for data association problem.

4.2.3 Data Association for EKF/UKF SLAM

No matter which model we choose, as long as it is a feature-based SLAM, it needs a reliable data association to update the state of the robotic platform and the tracked landmarks that form the map. The most naïve data association implemented for EKF/UKF SLAM is to use a single decision on a measurement based on the maximum likelihood with the estimated landmark position. If the map is formed solely by the static landmarks that are reasonably sparse, then the simple data association filter is sufficient to update the map by just incorporating the new measurements obtained from the current scan. However, if the scan area is cluttered with densely located landmarks or other objects that cause many false alarms, the simple data association based on a single ML decision is very brittle. Any single error in current data association can cause more errors

in the future data associations and results in the fatal consequence in EKF/UKF SLAM. In view of this, in next chapter, we investigate different algorithms from estimation theory used for tracking applications and tailor them to solve the problems of SLAM data association in the dynamic scenarios.

Nevertheless, recently more and more research literatures focus on SLAM models based on particle filter, such as FastSLAM, which is developed for the non-parametric multi-modal scenario. However, this kind of SLAM takes another approach for data association which is similar to the multiple hypotheses tracking (MHT) algorithm of tracking community [Nieto et al., 2003]. We also take effort to analyze its advantages and disadvantages in the following sections.

4.2.4 Non-parametric model: Particle Filter and FastSLAM

Particle filters, also known as Sequential Monte Carlo method is the basis for FastSLAM [Montemerlo and Thrun, 2007]. Similar to Markov chain Monte Carlo (MCMC) batch method, FastSLAM is also an importance sampling method. If well-designed, particle filters can be much faster than MCMC. They are often an alternative to the EKF or UKF with the advantage of being able to approach the Bayesian optimal estimate, if given with sufficient samples. So they can be made more accurate than either the EKF or UKF. The approaches can also be combined by using a version of the KF or EKF as a proposal distribution for the particle filter. The most advantage is that particle filter can be used for the situation when a Gaussian or an analytical model is not available. For example, suppose we know that some physical process is described by a random variable, but that we are not sure what this random variable looks like. The first piece of information that we want is the mean of this random variable. If we had the pdf for this random variable,

$p_x(x)$ we could directly compute the mean. If we do not have the pdf, we have to estimate the mean by collecting samples of the process and then take the average of these samples.

Since either MCMC or Particle filter have been well studied topics, or the related literatures are readily available in research community, we do not intend to further elaborate their principle and development in this thesis. The detailed description and derivation of particle filter can be found in references [Thrun, Fox, and Burgard, 2005].

FastSLAM is a hybrid filter model which utilizes particle filter for prediction stage and EKF for update stage. The most advantage is to be able to factor the conditionally independent landmark positions in the posterior distribution. This feature definitely reduces the processing time and complexity. FastSLAM can be represented as

$$\begin{aligned} & p(x_{1:t}, l_{1:m}) \mid z_{1:t}, u_{0:t-1}) = p(x_{1:t}) \mid z_{1:t}, u_{0:t-1}) \bullet p(l_{1:m}) \mid x_{1:t}, z_{1:t}) \\ & = p(x_{1:t}) \mid z_{1:t}, u_{0:t-1}) \bullet \prod_{i=1}^{M} p(l_i) \mid x_{1:t}, z_{1:t}) \end{aligned} \tag{4-17}$$

where N = number of particles (samples); M= number of features observed. The computational complexity is O (N log (M)).

There are two kind of FastSLAM, namely FastSLAM 1.0 and FastSLAM 2.0. The difference is their proposal distribution [Montemerlo et al., 2002]. FastSLAM 2.0 is assumed to be faster than FastSLAM 1.0 for its better estimation of proposal distribution.

4.2.5 Data association for particle filters and FastSLAM

Since the FastSLAM adopts the multi-hypothesis approach for prediction phase and each particle represents a different hypothetic path of the robot, so data association decision can be made on a per-particle basis. In this case, particles pick the correct data association will receive higher weights and thus have more chances to stay during the re-

sampling process. On the other hand, particles that choose the wrong data association will receive lower weights and will be marginalized later in future re-sampling process. Therefore the landmark filters in a single particle are not affected by the motion noise. This is similar to the optimal data association algorithm called Multiple Hypothesis Tracking (MHT). This algorithm enumerates the feasible measurement-to-landmark association hypothesis within a certain time-depth, and then evaluates the related probabilities and state estimates that are later combined into an overall state estimates of the landmarks. Besides, it uses a postponed decision logic which allows data association decisions to be delayed until additional scans of measurements are successively received so that incorrect associations made in the past can be revoked if necessary. In term of SLAM, MHT can be represented by a Gaussian mixture as shown below

$$Bel(x_t) = \frac{1}{\sum_l \psi_{t,l}} \sum_l \psi_{t.l} \left|2\pi\Sigma_{t.l}\right|^{-1/2} \exp\left\{-\frac{1}{2}(x_t - \mu_{t,l})^T \Sigma^{-1}(x_t - \mu_{t,l})\right\} \qquad (4\text{-}18)$$

where l is the index of mixture Gaussian with mean μ and covariance Σ. The scalar $\psi_{t,l} > 0$ is a mixture weight which determines the weights of l-th mixture Gaussian in posterior. However, the typical MHT algorithm is very computationally expensive due to the use of exhaustive search to enumerate all possible hypothesis and its complexity increases exponentially with the size of particle set. Take the case described in [Nieto et al., 2003] for example; each particle is split into {n + 2} particles, one for each of the n hypotheses, plus one particle for the non-association hypothesis (for spurious measurements) and one for the new landmark hypothesis. The computational complexity is up to O ($(n+2)\bullet m\bullet l$)

where m is the number of measurements and *l* is the number of tracking legs (tree of hypothesis). For the limited source on the small robots, it is not a feasible solution.

4.3 A quick survey of applicable data association algorithms

In the following table, we provide a glimpse of contemporary algorithms of data association from the tracking community [Daum, F. 1996]:

Table 4-2 Comparison of some tracking data association algorithms

Algorithm	Number of Data Association Hypotheses	Resolve Unassigned Data	Performance In dense environment	Computational Complexity
NN	1	No	Poor	Low
PDA	1	No	Poor	Low
JPDA	1	No	Fair	Medium
NNJPDA	1	No	Good	Low
Viterbi	Many	No	Good	Medium
MHT	Many	No	Optimal	High to poly[1]
Assignment	1	No	Good	Medium to poly
S-D Assignment	Many	No	Excellent	High
m-best S-D Assignment	Many up to m	No	Excellent	Medium
Morefield	Many	No	Good	high

[1] Problem can be solved in polynomial time in the sense of optimization estimation theory.

Chapter 5

Data Association for Dynamic SLAM

In this chapter, we begin to address the core problems of this research and investigate the relative approaches to solve the problems. As mentioned earlier, the data association is the key to the successful update of states for SLAM. Recall the Bayesian filter for SLAM, the measurement update model is based on the robot state $\mathbf{x}_k$, map **m**, and the observation measurements, $\mathbf{z}_k$. Now consider a scenario of cluster of targets (established landmarks) denoted as $\mathbf{X}_k = \{x_1, x_2 \ldots x_n\}$ at a given time k (For simplicity, the robot pose **x** is not included in the set). The set of measurements observed at this time-stamp is denoted as $\mathbf{Z}_k = \{z_1, z_2 \ldots z_m\}$, as shown in Figure 5-1(a) (the triangle represents measurements of a scan). Due to the uncertainty of the vehicle position and the sensor imperfection (error), the discrepancy between the estimate positions and the measurements from sensor observation is described by the innovation vector [Smith, Self, and Cheeseman, 1987]

$$\nu_m(k) = z_m(k) - \hat{z}_k \tag{5-1}$$

$$\hat{z}_k = h(\hat{x}_{k|k-1,} m) + \delta_k \tag{5-2}$$

where $\hat{z}_k$ is the estimate target position. The innovation covariance matrix is

$$S = \nabla h(\hat{x}_{k|k-1}) \Sigma_{v,k|k-1} [\nabla h(\hat{x}_{k|k-1})]^T + Q_k \tag{5-3}$$

where $\nabla h(\hat{x}_{k|k-1})$ is the Jacobian of the measurement function, and Σ_v is robot state

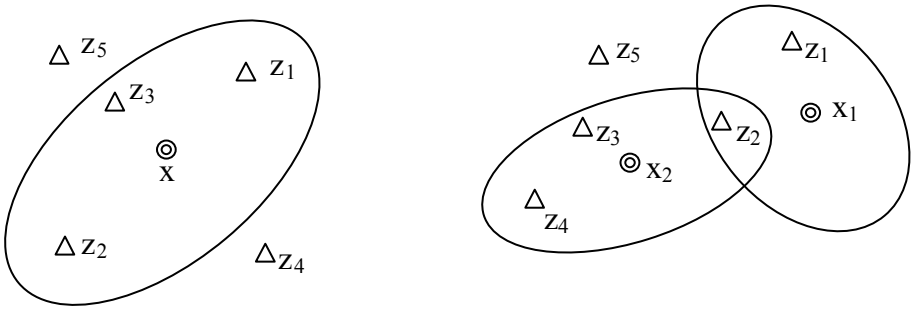


Figure 5-1. (a) Validation gate of single target. (b) A cluster of 2 overlapped gates

covariance. The smaller the innovation vector (ν), the more accurately the observation measurement matches with the tracked landmark.

The statistic validation region (gate) used to determine whether an observation measurement z_m is originated from this estimate target position x_n, is then defined as

$$M_x = v^T S^{-1} v \le \gamma_n \tag{5-4}$$

This is known as Mahalanobis Distance or NIS (normalized innovation squared). Note that γ_n is the gate threshold based on a Chi-squared (χ^2) distribution of n-dimension (see Appendix B). This is the standard criteria to test the validation of the received signal of observation. Among all the possible measurements, the one which is closest to the predicted position of the landmark will be selected. This is so called the nearest neighbor (NN) algorithm. However, if the validation gates (as shown in Figure 5-1(b), the ellipses) are overlapped, the data association based on such an assumption will fail to recover the true data association because the measurement inside the intersection of ellipses may originate from another target and it may have the shortest distance to this target. The situation is worse if encountered with the clutter.

A variant of NN can be used to resolve the conflicting data association in this scenario as in Fig. 5-1(b). This algorithm is usually formulated as the optimal assignment problem, in which the Mahalanobis distance (or NIS) for each validated measurement to landmark pair is computed and the optimal assignments are made by minimizing the sum of the Mahalanobis distance for all possible pairings, subject to maximizing the number of the total assignments. This technique is called global nearest neighbor (GNN) and its optimal assignment solutions can be found by many linear programming algorithms such as Kuhn-Munkres (Hungarian) [Bourgeois, 1971] and Auction [Bertsekas, 1990].

However, the solution of the optimal assignment problem imposes extensive computational burden, and therefore the sub-optimal solutions are always desirable in real-time implementation [Nagarajan, 1987, Reid, 1979, and Shalom et al. 1995]. As we mentioned earlier, there are many data association techniques used for SLAM and target tracking. In this study, we rule out the MHT algorithm since it is NP-hard and impractical for real-time application (except some situation using particle filter based SLAM like FastSLAM). Instead, we adopt the algorithms similar to PDA, JPDA, and their variants.

5.1 Probabilistic Data Association (PDA) and Joint PDA

As we mention earlier, both PDA and JPDA methods use weighted average of innovation for state update [Shalom and Fortmann, 1988]. That depends on the combined innovation vector as indicated below:

$$\tilde{\nu}^x = \sum_{m=1}^{M} \beta_m^x \nu_m^x \tag{5-5}$$

where β_m^x is the association probability and ν_m^x is the innovation vector as indicated in Eq.(5-1). The difference between the two methods is in the way of calculating and using the association probability β_m^x which is the probability that the measurement *m*, is originated from target *x*, and β_0^x, the probability that none of the measurements originated from target *x*. Since PDA assumes only one target in the scenario, it is not appropriate for the dynamic SLAM. Therefore the more sophisticated JPDA which considers all the contribution from all measurements within the validation region is more appropriate algorithm for this situation. The association probability for JPDA is given as

$$\beta_m^x = \sum_{\phi} P\{\Phi(k) \mid Z^k\}\, a_m^x(\Phi) \tag{5-6}$$

$$\beta_0^x = 1 - \sum_{m=1}^{M} \beta_m^x \tag{5-7}$$

where m = 1, 2... M; x = 0, 1, 2... N; and Φ(k) is the joint association event at the current time k,

$$\Phi(k) = \bigcap_{m=1}^{M} \phi_m^x(k) \tag{5-8}$$

where $\phi_m^x(k)$ is the individual association event {measurement m originated from target x}, and $\{a_z^x\}$ is the binary hypothesis association indicator between measurement z and target x within the validation matrix Φ(k) as shown below:

$$\Phi(\mathrm{k}) = [\,\phi_m^x(k)\,] = \begin{array}{c} \begin{array}{ccccc} x_0 & x_1 & x_2 & \dots & x_N \end{array} \\ \begin{pmatrix} 1 & a_1^1 & a_1^2 & \dots & a_1^N \\ 1 & a_2^1 & a_2^2 & \dots & a_2^N \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_M^1 & a_M^2 & \dots & a_M^N \end{pmatrix} \end{array} \begin{array}{c} \\ z_1 \\ z_2 \\ \vdots \\ z_M \end{array} \tag{5-9}$$

In this matrix, we can see the entries of first column are "1" that indicate the clutter can be associated with any measurement in the validation region.

5.2 Computation of Probability of Hypothesis Association

From (5-6), the probability of the joint event Φ(k), conditioned on all measurements up to current time k is given as

$$P\{\Phi(k) \mid Z^k\} = P\{\Phi(k) \mid Z(k), Z^{k-1}\} = \frac{1}{c}\, p\{Z^k \mid \Phi(k), Z^{k-1}\} p\{\Phi(k)\} \tag{5-10}$$

The normalized constant c is the joint prior density of the measurements, and it is the sum of all numerators over Φ(k). From (5-10), the joint probability density function on the right-hand side is defined as

$$p(Z^k \mid \Phi(k), Z^{k-1}) = \prod_{m=0}^{M} p\left[z_m(k) \mid \phi_m^x(k), Z^{k-1}\right] \tag{5-11}$$

Since a measurement m associated with the target x assumed to be a normal density function, we have

$$p\left[z_m(k) \mid \phi_m^x(k), Z^{k-1}\right] = \begin{cases} N(z_m; \hat{z}_m^x, S_m^x), & if \;\; a_m^x(\Phi) = 1 \\ A^{-1}, & if \;\; a_m^x(\Phi) = 0 \end{cases} \tag{5-12}$$

The normal density function is defined as

$$N(z_m; \hat{z}_m^x, S_m^x) = P_G^{-1}\left(2\pi S_m^x\right)^{1/2} \times \exp\left(\tfrac{1}{2}\tilde{\nu}_m^x (S_m^x)^{-1} (\tilde{\nu}_m^x)^T\right) \tag{5-13}$$

where P_G is the probability of the true measurements, m, falling in the validation gate of target x; and are innovation vectors and covariance, respectively. For measurements not associated with any target are assumed uniformly distributed in the surveillance region of area A as indicated in (5-12).

The second factor in (5-10) is the prior probability of joint event, in which the total number of false alarms is denoted as m_0. The number of correct measurements is given as $m_c = M - m_0$, where M is the total number of observed measurements within validation region. Therefore, the prior probability is

$$P\{\Phi(k)\} = P\{\Phi(k) \mid \delta(\Phi), \varphi(\Phi)\} P\{\delta(\Phi), \varphi(\Phi)\} \tag{5-14}$$

Since the same set of detected targets is given by the number of permutations of M measurements, the first term of (5-14) is:

$$P\{\Phi(k) \mid \delta(\Phi), \varphi(\Phi)\} = \left(P_{mc}^{M}\right)^{-1} = \frac{m_0}{M!} \tag{5-15}$$

where $\delta(\Phi)$ is the binary target detection indicator (1 or 0),

$$\delta_x(\Phi) = \sum_{m=1}^{M} a_m^x(\Phi) = \ \leq 1, \text{ for x = 1...N} \tag{5-16}$$

and φ(Φ) is the number of false measurements in the event. In order to define the relationship between the binary indicators, it is convenient to denote a binary indicator $\tau_m(\Phi)$ representing all targets associated with measurement *m* in the event Φ . Therefore

$$\tau_m(\Phi) = \sum_{x=1}^{N} a_m^x(\Phi), \text{ for m = 1,2, ...M} \tag{5-17}$$

$$\varphi(\Phi) = \sum_{m=1}^{M} [1 - \tau_m(\Phi)] \tag{5-18}$$

The second term of (5-14) is:

$$P\{\delta(\Phi), \varphi(\Phi)\} = \prod_{n=1}^{N} (P_D)_n^{\delta} (1 - P_D)_n^{1-\delta} P_F(m_0) \tag{5-19}$$

where p_F is Poisson probability density function for the false alarms:

$$p_F(m_0) = e^{-\lambda A} \frac{(\lambda A)^{m_0}}{m_0!} \tag{5-20}$$

where λ is the spatial density of false measurements (i.e. the average number per unit area) and *A* is the area of validation region (not a single gate). For a 2D elliptical validation gate, the area is represented by

$$A = \pi \mid \gamma S(k) \mid^{1/2} \tag{5-21}$$

where γ is chi-square pdf threshold and S(k) is the innovation covariance. Therefore, Eq. (5-14) becomes

$$P\{\Phi(k)\} = \frac{m_0!}{M!} e^{-\lambda A} \frac{\lambda A}{m_0!} \prod_{n=1}^{N} (P_D)_n^{\delta} (1 - P_D)_n^{1-\delta} \tag{5-22}$$

Measurements not associated with any target are assumed to be uniformly distributed in the observation area, A for a 2D scenario (V is the volume for 3D cases). The uniform density distribution for false alarms raised to the power of m_0 is given in [Shalom and

Fortmann, 1988; Neira, and Tardos, 2002]. By combining (5-12) and (5-13), Eq.(5-11) becomes

$$p\left(Z^k \mid \Phi(k), Z^{k-1}\right) = A^{-m_0} \prod_{m=1}^{M} N(z_m; \hat{z}_m^x, S_m^x) \tag{5-23}$$

Substituting (5-22) and (5-23) into (5-10) yields

$$P\left\{\Phi(k) \mid Z^k\right\} = \frac{\lambda^{-m_0}}{c} \prod_{m=1}^{M} N(z_m; \hat{z}_m^x, S_m^x) \prod_{n=1}^{N} \left(P_D\right)_n^{\delta} \left(1 - P_D\right)_n^{1-\delta} \tag{5-24}$$

This is the joint likelihood of associating a tracked target x with a measurement m. Recall the individual association probability from (5-6), using scenario of Fig. 5-2(b) for example, we are able to obtain the association probability for associating target x_1 with observation z_1 in a hypothetic joint event such as:

$$\beta_1^l = \sum_{\Phi} P\left\{\Phi(k) \mid Z^k\right\} a_1^l(\Phi) = p\left\{\phi(\Omega_1) \mid Z^k\right\} + p\left\{\phi(\Omega_2) \mid Z^k\right\} + p\left\{\phi(\Omega_3) \mid Z^k\right\} \tag{5-25}$$

Likewise, the association probability of other valid pair (when the binary association indicator is 1) can be computed as well. The total number of at each time step k is (M+1) N, where M and N are the number of measurements and landmarks respectively. However, some are too small to contribute to the weight which affects the joint innovation vector.

Finally, when substituting (5-6) into (5-5), the new state is for an EKF SLAM (if this is the SLAM algorithm currently used) can be updated as

$$x_{k|k} = x_{k|k-1} + K_k \tilde{\nu}_k \tag{5-26}$$

$$\Sigma_{k|k} = \Sigma_{k|k-1} - K_k S_k K_k^T \tag{5-27}$$

where x is state vector of both the robot and the landmarks, ∑ is the augmented process covariance, is the weighted innovation given in (5-5), S_k is the innovation covariance, and K_k is the Kalman gain and is given as

$$K_k = \Sigma_{k|k-1} H_k^T S_k^{-1} \tag{5-28}$$

where H_k^T is Jacobian of measurement function $h(\hat{x}_{k|k-1})$ as shown in (5-3).

5.3 Reduction of computation of JPDA

The computational complexity of generic JPDA described above may be exponentially high if the environment is crowded and the false alarm rate is high. In order to reduce the computation in the stressful environment, we follow the algorithm called "NNJPDA" (nearest neighbor JPDA) developed by R. Fitzgerald [Fitzgerald, 1990]. As we have mentioned earlier, the way NNJPDA makes the association decision is similar to GNN, except that the associating "cost" of GNN is the normalized innovation square (NIS) distance, while the associating "cost" of NNJPDA is the association probability. The computation of association probability is similar to Cheap JPDA [[Fitzgerald, 1985] given as

$$\beta_m^x = \frac{\Lambda_{m,x}}{\sum_{j=1}^{m} \Lambda_{m,x} + \sum_{x=1}^{N} \Lambda_{m,x} - \Lambda_{m,x} + B} \tag{5-29}$$

where $\Lambda_{m,x}$ is the likelihood of associating measurement *m* to target *x*, and B is a bias accounting for clutter and for the probability of detection less than 1. The likelihood function of the measurements from k-N up to time step k is given as

$$\Lambda_k = \prod_{i=k-2}^{k} \left| 2\pi S(i) \right|^{1/2} \exp\left(-\tfrac{1}{2} \sum_{i=k-2}^{k} \nu_i^T S_i^{-1} \nu_i \right) \tag{5-30}$$

where ν_i ,S_i are innovation vector and covariance respectively. The Cheap JPDA assigns a large association probability when B = 0. For non-zero B, it reflects that the measurement could be from a clutter or the target-originated measurement was not detected (miss). In our case, it can be adjusted according to the clutter density and the threshold can be empirically set when all the false targets are eliminated. In order to save the computation, Fitzgerald suggests using only three "best-fit" measurements for each target (with the highest association probabilities). After the association is done, remove this pair from further consideration. This process will be repeated until all the association decisions have been made.

5.4 One to One constrained Assignment Formulation used in JPDA

Since JPDA algorithm is formulated as the constrained optimization assignment problem [Wijesoma, 2004]. Assuming a scenario consisting of two true targets, clutter, and four measurements within a cluster as shown in Figure 5-1(b), there are four measurements observed in a cluster of two targets. Each measurement could have come from one of these targets or the clutter. The total number of feasibility event matrices from one validation matrix depends on the situation which gives rise to the constraint of that validation matrix. For the extreme case that all the measurements fall in the intersection of the validation regions of all targets, that is, all the entries inside the validation matrix are "1", then the total number of hypothesis can be computed as many as

$$\Omega_k = \sum_{j=0}^{\min(m,n)} \frac{M!N!}{(M-i)!(N-i)!\times i!} \tag{5-31}$$

Where M is the number of observations and N is the number of landmarks. The actual number is less than this, since in most cases the landmarks are not congregated. The more

targets are spread out, the less the number of feasibility hypothesis set is [Popp, 2001]. The feasible events are mutually exclusive, and they contribute to a certain target association probability with their weights. Based on this scenario shown in Figure 5-1(b), a validation matrix Ω is formed as follows:

$$\Omega = \begin{array}{cccc} x_0 & x_1 & x_2 & \\ \hline 1 & 1 & 0 & z_1 \\ 1 & 1 & 1 & z_2 \\ 1 & 0 & 1 & z_3 \\ 1 & 0 & 1 & z_4 \end{array} \rightarrow \{a_z^x\} = \begin{array}{ccccccc} & x_0 & x_1 & x_2 & \dots & x_N & \\ \hline & 1 & a_1^1 & a_1^2 & \dots & a_1^N & z_1 \\ & 1 & a_2^1 & a_2^2 & \dots & a_2^N & z_2 \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & 1 & a_M^1 & a_M^2 & \dots & a_M^N & z_M \end{array} \tag{5-32}$$

Target x_0 denotes the clutter or new target, while x_1 and x_2 are known targets. The binary association indicator a_z^x is "1" if measurement **z** lies inside the validation gate of target x, otherwise a_z^x is "0". Basically, this arrangement is telling us that we can only pick one element from a row except element from x_0 column. Similarly, select only one element from x_1 and x_2 columns.

Because the number of hypotheses of data association would grow exponentially with the increasing number of landmarks and measurements in case of dynamic SLAM, the crux left here is how to generate the feasible event set $\Phi(\Omega)$ without extensive and painstaking search effort. Since the 2-D assignment in the joint probabilistic data association is a process of matching the measurements in a scan to the list of established landmarks, and therefore it is similar to the classic application of the assignment problem in the field of constraint optimization theory. As such, it can be treated as a constrained optimization problem, except the parameters for calculating the cost and formulation of the constraints are different. One big advantage of using assignment optimization theory

for the JPDA hypothesis generation is the field of optimization is well established and there are plenty of optimizing problem solving algorithms available.

5.5 Formulation of Assignment Problem Algorithms and the solutions

The classic assignment problem example is formulated as follows:

Let i = {1…I} be a set of employee who are assigned to a set of task denoted as j = {1…J}. Each employee can be assigned to only one task at a time exclusively. There is a cost associated with each assignment. For example, c_{ij} is the cost of assigning employee I to task j. The constraint of assignment is also imposed as certain tasks are prohibited to be performed by certain employees if they do not have access authority. The natural formulation of this problem can be described as

$$\min C = \sum_{i=1}\sum_{j=1} c_{ij}x_{ij} \quad \text{(Minimize the objective function)} \tag{5-33}$$

$$s.t. \quad \sum_{i=1}\sum_{j=1} c_{ij}x_{ij} \le b_i \quad i \in I$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad j \in J \qquad \text{(Subject to the constraints)}$$

$$x_{ij} \in \{0,1\} \qquad i \in I, j \in J$$

The algorithm helps optimize the decision on which hypothetic set of assignment is optimal (the total cost C is therefore minimized!).

In our case, in order to formulate the joint hypothesis of feasible events within a cluster of the JPDA filter, we may define the binary assignment variable as $\mathbf{a}_{nm}$ instead of $\mathbf{x}_{ij}$ in order to be consistent with the context of this paper.

In the following chapter, we discuss the algorithms of linear programming for solving the assignment optimization problems. We are particularly interested in the integer programming methods that provide the more optimal solutions to the asymmetric (rectangular matrix) assignment problems.

Chapter 6

Assignment optimization for JPDA feasible hypothesis generation

The key idea of generic JPDA is that a set of measurements from current scan can be matched with a previous track file by formulating the match as a constrained global optimization problem [Kragel, B., 2010]. The one-one constraint assignment optimization problem can be solved by the linear (integer) programming algorithms [Wijesoma, 2004].

In this chapter, we first formulate our data association (correspondence between the landmark and measurement) as an optimized assignment problem with the constraint according to the special requirements given in our robotic SLAM scenario, not other applications. And then we investigate the appropriate solutions from several popular linear and integer programming algorithms developed for optimization estimation. These algorithms include Simplex, Branch and Bound, Auction and JVC (Jonker-Volgenant-Castenan) algorithm ...etc. The assessment of these methods has been conducted and we select to use Hungarian, Auction and JVC as major tools for our task of JPDA hypothesis generation.

6.1 Formulation of one-one assignment optimization problem for JPDA in SLAM

The optimization is done by minimizing the "cost" of associating the validated measurement to the targets in track file. It is a classic 2-D (2 scans in two-dimensional scan area) assignment problem of optimization in the field of estimation research. We can take advantage of abundant algorithms that have been developed in the past to solve this kind of problem. In the context of 2-D assignment, we first redefine the binary hypothesis association indicator between measurement z and target x as shown in the validation matrix Ω in Eq. (5-32) such that

$$a(k,m,n)=\begin{cases}1 & \text{if } z(k,m) \text{ is assigned to } x(k-1,n)\\ 0 & \text{otherwise}\end{cases} \quad \forall\, m = 0, 1, 2\ldots M(k);$$

$$n = 0, 1, 2 \ldots N(k\text{-}1) \tag{6-1}$$

where M(k) and N(k-1) are the cardinalities of the measurement set in current time-step and track set in the previous time-step, respectively. The indices $m = 0$ corresponds to dummy measurement and the assignment variable $\boldsymbol{a}$ (k, 0, n) denotes that the track n is not associated with any measurements in M(k). So we can call this a "miss" of detection. On the other hand, $n = 0$ corresponds to a dummy track and the assignment variable **a** (k, m, 0) denotes that the measurement is not associated with any existing track of N(k-1). We call this a "false alarm". The objective of assignment is to find the optimal assignment $\boldsymbol{a}^*$(k) which minimizes the cost of data association. It is expressed as follows:

$$\text{Min } C(k \mid a(k)) = \sum_{m=0}^{M(k)} \sum_{n=1}^{N(k-1)} a(k,m,n)\, c(k,m,n) \tag{6-2}$$

Subject to $\sum_{m=0}^{M(k)} a\,(k.m,n) = 1, \;\; \forall\; n = 1,\ldots,N(k-1)\,;\;\; \sum_{m=0}^{M(k)} a\,(k.m,0) \geq 0,$ (false alarm or new target)

$\sum_{n=1}^{N(k-1)} a\,(k.m,n) = 1, \;\; \forall\; m = 1,\ldots,M(k)\,;\;\; \sum_{n=0}^{N(k-1)} a\,(k.0,n) \geq 0,$ (miss of detection)

where c(k, m, n) is the cost of the assignment between the m measurement and the n track and given as

$$c\,(k.m,n) = -\ln\left(\frac{\phi(k,m,n)}{\phi(k,m,0)}\right) \tag{6-3}$$

$$\phi\,(k.m,n) = \begin{cases} P_D \Lambda(k,m,n) & \text{if } m > 0,\ n > 0 \\ A_s(k)^{-1} & \text{if } m > 0,\ n = 0 \\ (1-P_D) A_s(k)^{-1} & \text{if } m = 0,\ n = 0 \end{cases} \tag{6-4}$$

where P_D is the probability of detection; $A_s(k)$ is the 2-dimensional current scan area; and $\Lambda(k,m,n)$ is the likelihood function of the m^{th} measurements from k scan originated from n^{th} target as defined in (5-30).

6.2 Integer Linear programming algorithms to solve assignment problem

Among these algorithms, Hungarian (Kuhn-Munkres) is the classic method particularly designed to solve for assignment optimization problems. The advantage of Hungarian method is the simplicity and easy to implement. However, the canonical form of Hungarian method is to solve for symmetric assignment (square matrix) problems. For asymmetric (rectangular) assignment problem, it needs some special tweaking. An extension of this algorithm was introduced by Bourgeois et al. in [Bourgeois, 1971]. This extension of Hungarian method allows the algorithm to operate in situations where the numbers of assignees (targets) and tasks (measurements) are not the same, thus the name asymmetric (rectangular) matrix. In addition to Hungarian method, other algorithms such as Auction (or modified Auction) [Bertsekas, 1990] and JVC methods [Jonker et al. 1987] are also presented and evaluated as the comparison for performance.

6.2.1 Hungarian Algorithm for Asymmetric Assignment problem

The Hungarian algorithm is a combinatorial optimization algorithm which solves assignment problems in polynomial time, and has the complexity of O (n^3). The first version, known as the Hungarian method[1], was invented and published by Harold Kuhn in 1955. This was revised by James Munkres in 1957, and has been known since as the Hungarian algorithm, or the Kuhn-Munkres algorithm. The basic principles are stated as following:

[1] Harold Kuhn who gave the name "Hungarian method" because the algorithm was largely based on the earlier works of two Hungarian mathematicians: Kőnig and Egerváry.

Lemma 1: *Given a column vector (c_i) and a row vector (r_j), the matrix (b_{ij}) with the elements $b_{ij} = a_{ij} - c_i - rj$ has the same optimal assignment solutions the matrix (a_{ij}) .*

Lemma 2: *Given a matrix, (a_{ij}), if m is the maximum number of independent zero elements in this matrix, then there are m lines (row or columns or both) which contain all the zero elements of (a_{ij}).*

Definition: A set of elements of a matrix are independent if none of them occupies the same row or column.

Lemma 3: *If a number is added to or subtracted from all of the entries of any one row or column of a cost matrix, then an optimal assignment for the resulting cost matrix is also an optimal assignment for the original cost matrix.*

We adopt the extended Munkres algorithm from [Bourgeois, 1971] to solve assignment problem in the following steps:

Step 1: For each row of the matrix, find the smallest element and subtract it from every element in this row.

Step 2: Find a zero of the matrix. If there is no zero with "*****" mark in its row or its column, mark it with "*****". Repeat for each element in the matrix. Then go to step 3.

Step 3: Cover every column containing a "0*". If all columns are covered, this means a complete set of unique assignments; optimal solution is obtained, then Exit and Done. Otherwise, go to step 4.

Step 4: Choose an uncovered zero and prime it (**0'**). If there is no "0*" in this row containing "**0'**" , go to step 5. If there is a "0*" in this row, cover this row and uncover the column containing "0*". Repeat until all zeros are covered. Go to step 5.

Step 5: Build a sequence of alternating "**0***" and "**0'**" as follows: let Z_0 denote the uncovered 0'. Let Z_1 denote the 0* in the column of Z_0 (if any). Let Z_2 denote the 0' in the row of Z_1. Continue in a similar way until the sequence stops at a 0', which has no 0* in its column. Remove the "0*" of the sequence, and put "*" on each primed zero of the sequence. Erase all primes and uncover every line in the matrix. Return to step 3.

Step 6: Add the value h found in Step 4 to every element of each covered row; then subtract h from each uncovered column. Return to step 4 without altering any asterisks, primes, or covered lines.

Result interpretation: Assignment pairs are indicated by the positions of "0*" in the cost matrix. The Hungarian algorithm can be demonstrated by the following example using the scenario of Fig.5-1 (b) and assume the validation matrix in (5-32) with the cost of assignment is given as follows:

$$a(k,m,n)c(k,m,n)=\begin{bmatrix}1&1&0\\1&1&1\\1&0&1\\1&0&1\end{bmatrix}\bullet\begin{bmatrix}2&3&x\\4&2&5\\7&x&4\\2&x&8\end{bmatrix}\Rightarrow\begin{bmatrix}2&3&\infty\\4&2&5\\7&\infty&4\\2&\infty&8\end{bmatrix}\Rightarrow\begin{bmatrix}0&1&\infty\\2&0&3\\3&\infty&0\\0&\infty&6\end{bmatrix}\Rightarrow\begin{bmatrix}0^*&1&\infty\\2&0^*&3\\3&\infty&0^*\\0&\infty&6\end{bmatrix}$$

(Note: set the unrelated cost to infinity which means most unlikely pairs due to the negative log-likelihood). Since k = min (row, column) =3, and according to Step 3, we can obtain the optimal solution for this easy case. But it is not always the case.

Extension of algorithm can be implemented as follows:

(a) k = min (n, m), no lines are covered; no zeros are starred or primed.

(b) If the number of rows is greater than the number of columns, go at once to step 1. Consider a particular row of the matrix (a_{ij}); subtract the smallest element from each element in the row; do the same for all other rows. If the number of columns is greater than the number of rows, go at once to step 3.

(c) Another simple way to change symmetric matrix (square) to asymmetric matrix (rectangle) is to pad the extra rows (or columns) with zeros. But it may be slower than (b).

6.2.2 Auction algorithm for 2-D assignment problem

The basic concept of auction is that each person wants to be assigned to an item that minimizes his cost (or maximize his profit), and thus puts an amount to bid for it. Therefore the target (landmark) is analogous to the bidding person indexed by row, and the sensor measurement is analogous to the item indexed by column as shown in (5-32). The target (person), n, bids for the measurement (item), m, with a negative log-likelihood (price), and the net profit that target, n, can make is the difference between the value of item and the bidding price. Note that a target earns no profit if it assigns itself to dummy measurement, and the negative profit is never considered in the bidding process. In order to use the auction algorithm in 2-D assignment problem to generate a feasible solution, it has to be modified such that, instead of minimizing the objective function in (6-2), we maximize the objective function (the primal) as follows:

$$\text{Max } C(k \mid a(k)) = \sum_{m=0}^{M(k)} \sum_{n=1}^{N(k-1)} a(k,m,n)\, d(k,m,n)$$

where $d(k,m,n) = -c(k,m,n)$ and $c(k,m,n)$ is the cost function shown in (6-3)

The dual of this problem is given as

$$\min_{q_n, p_m} \left\{ \sum_{n=1}^{N(k-1)} q_n + \sum_{m=1}^{M(k)} p_n \right\}$$

subject to

$$q_n + p_m \geq d(k.m,n); 1 \leq n \leq N(k-1), 1 \leq m \leq M(k)$$

$$q_n > d(k,0,n) \;\&\; p_n > d(k,m,0)$$

Where q_n and p_n are the Lagrangian multipliers. This method is sometimes called "Jacobi- like Relaxation" method [Bertsekas, 1987].

6.2.3 JVC (Jonker-Volgenant-Castenan) algorithm for 2-D assignment problem

JVC algorithm resembles a special variation of Dijkstra's shortest path linear programming algorithm. It also has the primal and dual formulation similar to the auction algorithm. However, the number of tracked targets is made equal to the number of measurement by padding them with dummy targets or measurements. In addition, the row index, if $x_n = 0$, it corresponds to an unassigned row, likewise $z_m = 0$ represents an unassigned column. There are three steps in JVC algorithm, namely, initialization, augmentation, and dual solution update. The complexity of general JVC is about $O(n^3)$. We apply this algorithm in later chapter of this paper to improve the performance for optimized ranked assignment in the more complicate situation. Details for Auction and JVC algorithm also can be found in [Bertsekas, 1990], [Jonker, 1987] respectively.

6.2.4 Discussion of performance between Hungarian and other methods

Hungarian method may not yield the optimal assignment if there are many un-assignable pairs due to the false measurements (i.e., the last row in above example assignment matrix), unless the minimum cost of assignment can be raised to relax the tight constraint. This results in a sub-optimal assignment. Besides, it suffers the speed which is slowed down by sequential processing of the algorithm. To rectify this shortcoming, other algorithm such as Auction (or modified Auction) [Bertsekas, 1990] and JVC methods [Jonker et al. 1987] are proposed and have been proved to outperform the Hungarian algorithm in term of efficiency and optimality of problem solving in polynomial time.

The time complexity for Auction method is $\boldsymbol{O}$ (n^3C), where C is the range of the

cost coefficients; n is the number of landmarks to be associated with the measurements. With Jacobi-like relaxation and scaling method, the complexity is approximately O (NA log (NC)) where N is the number of persons, A is the number of pairs of persons and objects that can be assigned to each other, and C is the maximum cost of object. But for most scenarios, the typical complexity is close to $\boldsymbol{O}$ (n^2) if it is used in 2-D assignment problem. The complexity of using JVC in 2-D assignment problem is roughly $\boldsymbol{O}$ (n^3) [Miller, 2001]. Figure 6-1 is the run-time comparison between Auction and JVC using the same matrix of 100 by 100 entries. We may find JVC is much faster than Auction algorithm. When augmenting the same matrix by adding one more column to make it a rectangle matrix for running JVC, the run-time is still faster than Auction algorithm.

However, according to [Popp, 2001], Auction is more suitable for the problem of sparse environment, while JVC is more superior in case of dense environment. Therefore, we adopt JVC as our major linear programming tool to solve for the ranked assignment problem in the next chapter.

```
Auction-square:
Elapsed time is 1.485000 seconds.
JVC-square:
Elapsed time is 0.015000 seconds.
use 100x100 square matrix to test both JVC & Auction:
JVC-rectangle:
Elapsed time is 0.016000 seconds.
use 100x101 rectangle matrix to test JVC:
>>
```

Figure 6-1. Benchmark comparison between JVC and Auction.

Chapter 7

New target initialization and maintenance in clutter

As we state in chapter one, a successful data association largely depends on how it initializes the new target. In a cluttered environment, the real landmarks have to compete with the false targets. The false alarms from the clutter are characterized as spurious signals and having Poisson density function by nature. Although regular JPDA may be able to handle this situation and provide an acceptable performance in data association, as we mentioned earlier, it may not be able to tell the difference between the interfering moving objects and a real new targets. As such, the single scan JPDA may not provide reliable performance if operating in highly dynamic environment [Shalom and Fortmann, 1988; Popp, 2001]. More specifically speaking, the single scan assignment decisions may not be able to correctly initialize a new landmark due to the interference from the moving objects. Therefore, in difficult data association situations, postponing final decisions may be the best thing to do until more information, such as the data from next scan(s), are received.

Although we could use many other criteria, a cost function based on the joint likelihood between features and measurements over multiple time-frames can be used to boost the likelihood for stationary targets. In doing so, even though they might be falsely initialized to begin with, it still can be corrected by multiple scan data association scheme based on the assumption that the flickering measurements from the moving objects may have smaller congregated likelihood because they would have moved away from their current positions eventually. Based on this inference, the moving objects may be pruned out in multi-scan JPDA. The same inference is also assumed by the particle filter-based

SLAM data association in which the pruning is done by the recursive process of re-sampling and marginalizing the unlikely particles.

7.1 Sliding window 3S-2D algorithm for new target initialization

The sliding window is a very common algorithm to process the multiple scan tracking problem. We use three most recent scans, including k-2 scan, k-1 scan, and the current k scan. The joint likelihood for each scan can be expressed as

$$\Lambda(\Omega) = \Lambda_T(\Omega) \times \Lambda_F(\Omega) \tag{7-1}$$

where $\Lambda_T(\Omega)$ and $\Lambda_F(\Omega)$ are the joint likelihoods of measurements associating with true features and spurious measurements, respectively, in the event partition Ω . This is similar to joint likelihood of associating a tracked target x_n with a measurement z_{m} or z_0, as previously obtained in (5-24) which is restated here for the sake of reading continuity.

$$\Lambda(\Omega) = P\{\Phi(k) \mid Z^k\} = \frac{\lambda^{-m_0}}{c} \prod_{m=1}^{M} N(z_m; \hat{z}_m^x, S_m^x) \prod_{n=1}^{N} (P_D)_n^{\delta} (1 - P_D)_n^{1-\delta} \tag{7-2}$$

Since the number of true measurements and false measurements taken in three scans are different and they are assumed to be independent of one another; thus, the accumulated joint likelihood of 3 scans can be expressed as

$$\Lambda^{3S} = \prod_{s=k-2}^{k} \Lambda^S(\Omega) = \Lambda^{k-2}(\Omega) \times \Lambda^{k-1}(\Omega) \times \Lambda^k(\Omega) \tag{7-3}$$

and the negative log-likelihood of (7-3) is used to calculate the cost function for assignment optimization problem presented in chapter 6. Thus, the cost function is

$$\begin{aligned} C^{3S} &= -\log(\Lambda^{3S}) = \sum_{s=k-2}^{k} -\log\left[\Lambda^S(\Omega)\right] \\ &= -\log\left[\Lambda^{k-2}(\Omega)\right] - \log\left[\Lambda^{k-1}(\Omega)\right] - \log\left[\Lambda^k(\Omega)\right] = c^{k-2} + c^{k-1} + c^k \end{aligned} \tag{7-4}$$

where $c^{k-2} = -\log\left[\Lambda^{k-2}(\Omega)\right], c^{k-1} = -\log\left[\Lambda^{k-1}(\Omega)\right]$, and $c^k = -\log\left[\Lambda^k(\Omega)\right]$.

This congregated cost function can be embedded into the objective function (6-2) to formulate our 3S-2D assignment problem as follows:

$$\text{Minimize} \quad \begin{aligned} C(S \mid a(k)_{k-2}^{k}) = & \sum_{m(k-2)=0}^{M(k-2)} \sum_{n(k-3)=1}^{N(k-3)} a(k-2, m_{k-2,} n_{k-3}) c(k-2, m_{k-2,} n_{k-3}) \\ & + \sum_{m(k-1)=0}^{M(k-1)} \sum_{n(k-2)=1}^{N(k-2)} a(k-1, m_{k-1,} n_{k-2}) c(k-1, m_{k-1,} n_{k-2}) \\ & + \sum_{m(k)=0}^{M(k)} \sum_{n(k-1)=1}^{N(k-1)} a(k, m_{k,} n_{k-1}) c(k, m_{k,} n_{k-1}) \end{aligned} \tag{7-5}$$

Subject to the constraints similar to those shown in (6-2), except the assignment indicator $a(k,m,n)$, can be relaxed from integer variables {0, 1} to any non-negative number such that $a(k,m,n) \geq 0$. The threshold can be obtained from the prior weighted average of the likelihood of true targets of three scans.

7.2 Maintenance & Deletion by m out of n detections

Once a new track has been established as a legitimate landmark, it is supposed to be a permanent global landmark for the rest of SLAM processing. However, for a highly dynamic scenario, we need one more layer of confirmation to maintain the newly initialized landmarks. For this matter, we use a scheme, called "m out of n detections" to determine whether the joint likelihoods are true targets or false targets. This can be done by setting a likelihood threshold ratio (m/n), for example, 2 out of 3 times above the threshold; then it is considered a new legitimate target; otherwise, it is a false alarm and subject to be deleted from the landmark file. The procedure of initialization and maintenance is described in the following steps:

Table 7-1: The pseudo code of m out of n detections in 3S-2D sliding window

1	Calculate the accumulated negative log likelihood as in Eq. (7-3) & (7-4)
2	Calculate the threshold level by averaging the likelihood of false alarms of 3 scans
3	If the joint likelihood is greater than the threshold, then queue the measurement as a tentatively initialized landmark; else just ignore it.
4	Examine each tentatively initialized landmark in the queue by m/n detector
5	If they pass the threshold twice in 3 scans, then register them as permanent targets.
7	Otherwise, delete it from the landmark queue

7.3 Rescind the wrong data association decision by Q-best ranked assignments

No matter how carefully we plan to prevent wrong data association or wrong landmark initialization, in the dynamically changing environment, it is inevitable to encounter the unpredictable situation. For example, a moving object (such as person) stands still during many scan periods and is recognized as the stationary landmarks. After three scan (we use 3 scans in this work to save time; however, it is not limited to only 3 scans), its likelihood exceeds the threshold and it is therefore registered as a legitimate landmark. But later on, this object moves away. The location no longer has a landmark there. The wrong data association based on this landmark would cause error. As the errors continue to accumulate, it would results in large tracking error. For EKF SLAM, it manifests as a loop enclosure problem; for particle filter SLAM, it causes problem of convergence. Therefore, the ability to rescind the decision on data association is very important to SLAM. One way to remedy this is to retrieve the Q-best assignment set from the previous assignment processes to substitute for the current assignment.

Based on the proposal by [Murty , 1968; Miller, 2001; Popp, 2001], the algorithm is not only to find the single best solution, but also to find the second best, the third best, and down to the Q-best solutions, as the alternative to revert the previous decision on the optimal assignment of data association (i.e., Q can be any positive integer).

In order to formulate the Q-best assignment problem, we denote a feasible solution of assignment as

$$A_i(k) = \left\{\left\langle n, m, c_{knm} \right\rangle\right\} \bigcup \left\{\left\langle 0, m_0, 0 \right\rangle\right\} \bigcup \left\{\left\langle n_0, 0, 0 \right\rangle\right\} \tag{7-6}$$

where $\left\langle n, m, c_{knm} \right\rangle$ is the feasible event of associating measurement m to target n with cost $c_{k,m,n}$; $\left\langle 0, m_0, 0 \right\rangle$ is the event of associating the measurement to the false alarm with cost 0;

$\langle n_0,0,0 \rangle$ is the event of missed detection with no cost. The entire feasible solution space can be defined as follows:

$$A(k) = \bigcup \{A_i\} \; ; \{A_i\} \neq \{A_j\}, \; i \neq j \tag{7-7}$$

The size of A(k) or the number of all data association assignments is given in (5-31). The cost of (or likelihood) of an assignment, denoted by $C(A_i)$, can be found by summing the individual costs in (7-4) as

$$C(A_i) = \sum_{\langle m,n,c \rangle \in A_i} c(k, m_k, n_{k-1}) \tag{7-8}$$

To determine the Q-best 2-D assignments problem in polynomial time, we formulate the processing by using Murty's algorithm to rank the solutions in order of increasing cost. The 2-D assignments to the problem Ψ, i.e., $A^*_{(1)}$, $A^*_{(2)}$ ... $A^*_{(Q)}$ are the Q best assignments with the least costs, i.e.,

$$A_1^* = \arg \underbrace{\min}_{A_i \in A} \{C(A_i)\} \tag{7-9}$$

$$A_2^* = \arg \underbrace{\min}_{A_i \in A \setminus A_1^*} \{C(A_i)\} \tag{7-10}$$

$$\vdots$$

$$A_Q^* = \arg \underbrace{\min}_{A_i \in A \setminus A_n^*} \{C(A_i)\} \qquad \text{For all } A_n^*; \quad n = 1 \ldots Q\text{-}1 \tag{7-11}$$

where $A_i \in A \setminus A_n^*$ denotes that the subset in A but not in A_n^*. This algorithm solves a series of 2-D assignment problems where the original problem Ψ is partitioned into a number of sub-problems, say Ψ_i, i =1 ... Q, having solution spaces A_i, $A_i \subset A$, by removing the optimal assignment from Ψ_i successively . The partitioning task in Murty's algorithm maintains the following two constraints:

$$\bigcup_{n=1}^{Q} A_n = A - A_1^* \tag{7-12}$$

$$A_i \cap A_j = \emptyset, \text{ for } i, j = 1 \ldots Q, i \neq j \tag{7-13}$$

After partitioning Ψ according to its optimal assignment A_1^*, we solve each sub-problem Ψ_n, $1 \leq n \leq Q$, and pair it together with its optimal solution A_n^*, and store each pairing $\langle \Psi_n, A_n^* \rangle$ on a queue (a tentative track file). A_2^*, the 2nd best assignment to Ψ, becomes the optimal assignment corresponding to sub-problem Ψ_2 on the queue and has the minimum cost (or maximum likelihood) as shown in (7-10). To find the 3rd best assignment in Ψ, we simply replace sub-problem Ψ_n corresponding to the 2nd best assignment A_2^* by its partitioning in the queue. The optimal assignment, A_3^* corresponding to sub-problem Ψ_m, for $1 \leq m \leq 2Q$, in the queue that has minimum cost after partitioning Ψ_m would then be the 3rd best assignment to Ψ, and so on. Therefore, the tentative queue should contain a list of assignment pairs: $\langle \Psi_1, A_1^* \rangle$, $\langle \Psi_2, A_2^* \rangle$, $\langle \Psi_3, A_3^* \rangle$... and $\langle \Psi_Q, A_Q^* \rangle$.

Since we perform one partitioning task for each of the Q-best 2-D assignments, in the worst case, each partitioning creates N new problems. This creates up to Q×N assignment problems to be solved and insertions of $\langle$*problem, solution*$\rangle$ pairings on the queue of the track file. Solving each 2-D assignment problem has the worst case complexity, $O(N^3)$, and each insertion step has the worst case complexity O(QN). Hence, the worst case complexity of Q-best 2-D is O (QN (N^3 +QN)). [Miller & Popp, 2001; Leung, 1999].

The key computational cost of Murty's algorithm, as well as any assignment optimization methods, is the determination of the least cost assignment. Miller, et al. [Miller, 2001] proposed three optimizations to improve Murty's method, they are summarized as follows:

1) During the partitioning process, have the created sub-problems inherit dual variables and partial solutions from the initial problem. 2) After the partitioning process, sort the sub-problems by lower cost bounds before solving the assignment problems. 3) Partition in an order based on lower cost bounds.

The first optimization, inheriting dual variables and partial solutions during partitioning, is a well-known JVC optimization technique which reduces the worst case Complexity of Murty's algorithm from $O(QN^4)$ to $O(QN^3)$. The JVC algorithm for solving Murty's assignment problem can be outlined as follows:

1. Use a simple computation to produce a partial solution for the problem and initialize the dual variables.
2. While the current partial solution is incomplete (not all measurements are assigned to landmarks), then
   2.1. Add a new assignment to the solution – augmentation.
   2.2. Adjust the dual variables.

Recall from (6-2), the linear assignment problem is formulated to minimize the cost function of $\sum_{m=0}^{M(k)}\sum_{n=1}^{N(k-1)} a(k,m,n)\,c(k-1,m,n)$, which is subject to the following constraints: $\sum_{m=0}^{M(k)} a\ (k,m,n)=1,\ \ \forall\ n=1,...,N(k-1)$, and $\sum_{n=1}^{N(k-1)} a\ (k-1,m,n)=1,\ \ \forall\ m=1,...,M(k)$ for $a\ (k.m,n)\geq 0$. Then the dual problem is given as

$$\text{Maximize } \sum_{m} u(k,m)+\sum_{n} v(k-1,n) \tag{7-14}$$

Subject to

$$c(k,m,n)-u(k,m)-v(k-1,n)\geq 0\ \ \forall\ m=1,...M;\ n=1,...N \tag{7-15}$$

where *u(k,m)* and *v(k-1,n)* are the dual variables for *z* and *x*, respectively, and the *c(k,m,n) -u(k,m) - v(k-1,n)* is called the "slackness" of the assignment program.

Table 7-2: Murty's ranked assignment scheme by using JVC algorithm	
1	Find the optimal assignment, A_0^* to the problem, Ψ_0
2	Initialize the queue with problem/solution pair, $\{\Psi_0, A_0^*\}$, the top pair should have the lowest cost.
3	For I = 1 to Q or until the queue is empty
4	Remove the top pair from queue
5	Add A to the list of solutions to be returned
6	For each triplet, <x, z, c>, found in A
7	let $\Psi^1 = \Psi$; remove the triplet <x, z, c> from Ψ^1
8	Find the best solution, A_1^*, to Ψ^1
9	If Ψ^1 exists then
10	Add $\{\Psi_0, A_0^*\}$ onto the queue of problem/solution pair
11	Repeat until it reaches the upper bound of minimum cost

The first feature of this optimization is that it does not need to perform more than one augmentation for each new sub-problem after the first solution to the original assignment problem is found. The second optimization is the sorting sub-problems by Lower Cost Bounds before solving sub-problems. These bounds can be used to avoid solving sub-problems that are unlikely to produce the next best result. In fact, no sub-problem that results from partitioning a given problem Ψ can have a best solution better than the solution to Ψ itself. Thus the initial cost C_* of the solution to Ψ is an initial lower bound on the cost of the best solution to each of its sub-problems. When an individual sub-problem Ψ_{sub} is created by removing a triplet $\langle x, z, c \rangle$ from a copy to Ψ, a more accurate lower bound can be obtained by finding the minimum slackness of all the alternative assignments for z_i, and adding that slackness to C_*.

Table 7-3: Minimal Slackness to relax the tight constraints	
1	Let minSlack = ∞ (infinity)
2	For each assignment triplet $\langle x, z, c \rangle$, $i \neq j$ in Ψ_x
3	Slackness = $c_{ij} - u_i - v_j$
4	if slackness < minSlack; minSlack = Slackness
5	$C_{sub} = C_* +$ minSlack

The cost of the best solution to Ψ_{sub} cannot be less than C_*. Although it does not improve the worst case complexity, however it does improve the general case complexity. For example, when a problem is partitioned, lower cost bounds on the cost of the best solutions to its sub-problems can be determined. These bounds can then be used to avoid solving sub-problems that are unlikely to produce the next best assignment. The third optimization is to change the order in which the triplets are used for partitioning, thereby increasing the probability that smaller problems are more likely to contain the best solutions [Popp, 2001]. These optimizations have been incorporated into Q-best 3S-2-D for new target (landmark) initialization and maintenance.

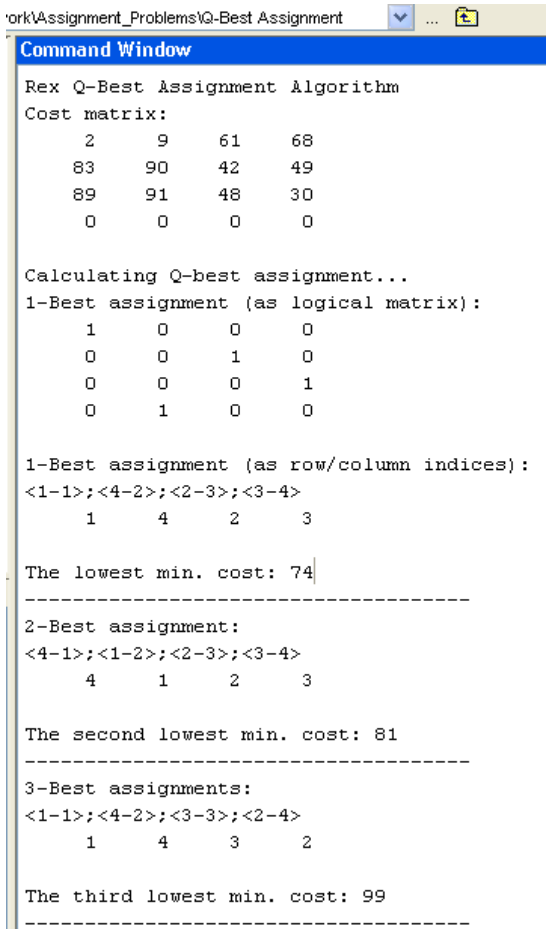


Figure 7-1 Example of Q-best solution

Figure 7-1 shows the result of running a linear programming algorithm for Q-best ranked assignment problem. Note that the cost matrix is padded with zeros for asymmetric assignment. The cost matrix is the example of cost functions that are calculated from the negative log likelihood C*ij*, between measurement *i* and landmark *j*. In this example, index *i* is the row index of landmarks and index *j* is the column index of measurements.

7.4 Adaptive validation gating

Because the Mahalanobis Distance (or NIS) involves matrix-vector and matrix-inversion computations for each measurement to determine their validity, the computational cost increases in heavily cluttered area. To alleviate the computational burden, a preliminary gate can be implemented to prune out the unlikely measurements. This can be done by using a simple threshold to form a rectangular gate as the first layer of measurement to track pairing selection

$$| z_m(k) - \hat{z}_m(k \mid k-1) | \leq \gamma \sqrt{S_m(k)} \tag{7-16}$$

where $S_m(k)$ is the innovation covariance of measurement m at time-step k. In this layer, any residual greater than this criteria is eliminated. Then the remaining ones will be further evaluated by the second layer gating of validation. However, the fixed gate threshold, γ_n, is not reflecting the change of dynamic situation in terms of target density, increasing false alarms, and interference of moving objects. In order to precisely validate the measurements and reduce the processing computation, a correlation between the additional parameters such as probabilities of detection and false alarms and the validation gates should be addressed as the dependence between these parameters should be quantified in a more systematic way. Furthermore, the result of tracking process (data association) should be able to feedback to the signal processing unit in helping reset the detection threshold for feature extraction. This shall form a complete feedback system adaptive to the dynamics of the changing environment, as shown in Figure 7-2.

We adopt from Fortmann's work [Fortmann et al., 1983, 1988] to derive the dependence of a tracker's error covariance upon detection and false alarm probability is explicitly (but approximately) characterized by a scalar parameter **q** in the covariance

equation (modified Riccati equation). The scalar parameter depends upon the probabilities of detection and false alarm, and also upon the volume of the data association gate, which in turn depends on the state error covariance matrix Σ_v**.** The modified Riccati equation can be iterated to convergence, yielding a steady-state Σ_v**,** and tracking performance can be characterized by a few scalar metrics such as determinant(Σ), trace (Σ) , **or** (in surveillance applications) root-mean-square position error. The derivation of **q** is given in Appendix C.

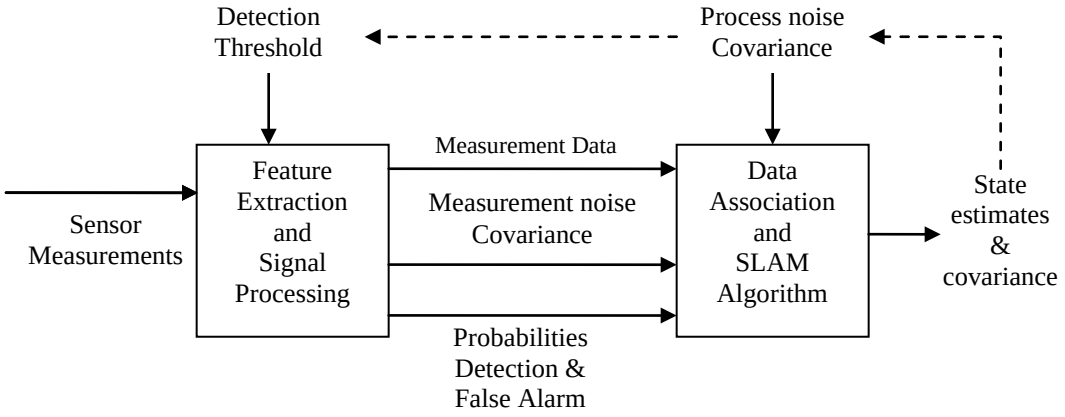


Figure 7-2. Complete tracking system block diagram.

When taking this data association for SLAM to the next level of complexity, that is, working with moving objects that may have some maneuver action, we cannot rely on a single dynamic process model based on some fixed estimation of the dynamic state variables. On the contrary, we opt to adopt the concept of multiple models for tackling the dynamic situation in the next chapter.

Chapter 8

Interactive Multiple Models

The real challenge of data association comes from the moving objects with some degree of maneuvering action. In many tracking applications, maneuvering action of a target can break the "lock" and cause track loss. For SLAM, the maneuvering objects cause the confusion in data association between the measurements and landmarks. A tracking estimator with single set of parameters most likely cannot handle this situation. Only the estimator with multiple set of parameter can accommodate the various motions of the targets. This requirement directs our attention to the multiple model estimators.

8.1 The IMM data association system

The Interactive Multiple Model (IMM) is a state estimator for a system represented by Markovian switching scheme [Blom and Shalom, 1988]. It is sometimes called jump Markov or hybrid-state estimator because it involves both the continuous state processes and the discrete mode (model) variables. It has been used by other tracking communities for handling target with maneuvering motions. Recently the researchers in SLAM community also adopt the idea of IMM for the benefit of drift-free imaging based SLAM [Civera, Davidson, Montiel, 2008].

The IMM consists of a group of hypothesized models, say, M_1, M_2, M_3 ... and M_r, with a pre-determined probability of switching , P_{ij} between models. Over a scan period, only one model Mj of the entire model bank $\{M_1 \ldots M_r\}$ is in effect (it is also called the regime model). For a non-linear system, which is normally used in SLAM, the motion dynamics and the measurements with additive noise can be modeled by the following state equations:

$$x(k) = g^j(x_{k-1}, u_k) + \varepsilon_{k-1}^j \tag{8-1}$$

$$z(k) = h^j(x_k, m_k) + \delta_k^j \tag{8-2}$$

where ε(k-1) and δ(k) are filter process and measurement noise, with zero-mean and random white Gaussian covariance, Q(k-1) and R(k), respectively.

The model M_j is in effect (in regime) during the scan-sampling period (from t_{k-1} to t_k). The vector x(k) is the system state at time k; z(k) is the measurement vector at time k; $g^j(x_{k-1}, u_k)$ is the non-linear state transition matrix for model j from time t_{k-1} to t_k;

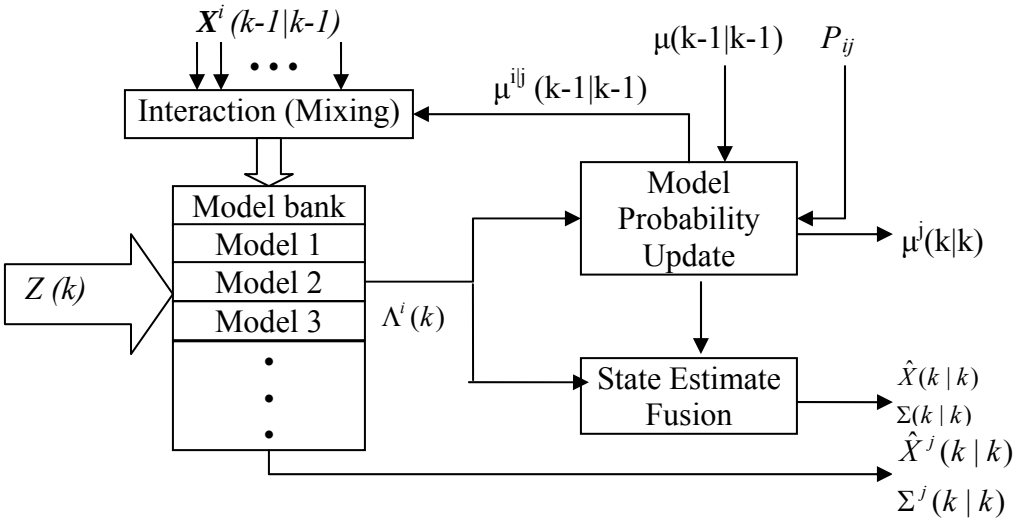


Figure 8-1. System Block Diagram for IMM estimator

and $h^j(x_k, m_k)$ is the observation matrix for model M_j at time k. The process noise ε_k and the measurement noise δ_k are uncorrelated zero-mean white Gaussian random processes with covariance matrices Q_k^j and R_k^j, respectively.

At the initial time zero t_0, the initial conditions of system state of each model M_j are assumed to be Gaussian random variables with mean $\bar{x}_0^j$ and covariance Σ_0^j, as well as the initial probability of choosing the correct model, $\mu_0^j = \Pr\{M_0^j\}$, which is assumed known. The model switching mechanism is governed by a finite state Markov chain

according to the probability, P_{ij} = Pr $\left\{M_k^j \mid M_{k-1}^i\right\}$ which is the probability of switching from M_{k-1}^i to M_k^j. It can be tuned based on certain pre-set parameters according to the mission requirements and scenario. More in details can be found in reference [Shalom, Li, and Kirubarajan, 2001]. The operation of IMM is summarized as follows:

1. Interaction (mixing) of the State Estimation. For the filter matched to the model Mj(k), the mixed estimate and covariance input to the IMM filter are:

$$\hat{x}^{0j}(k-1\mid k-1)=\sum_{i=1}^{r}\mu^{i\mid j}(k-1\mid k-1)\hat{x}^{i}(k-1\mid k-1) \tag{8-3}$$

$$\Sigma^{0j}(k-1\mid k-1)=\sum_{i=1}^{r}\mu^{i\mid j}(k-1\mid k-1)\{\Sigma^{i}(k-1\mid k-1)+[\hat{x}^{i}(k-1\mid k-1) \tag{8-4}$$

$$-\hat{x}^{0j}(k-1\mid k-1)][\hat{x}^{i}(k-1\mid k-1)-\hat{x}^{0j}(k-1\mid k-1)]^{T}$$

where the conditional model mixing probabilities, $\mu^{i|j}$(k-1|k-1) are given by

$$\mu^{i\mid j}(k-1\mid k-1)=P\left\{M^{i}(k-1)\mid M^{j}(k),Z_1^{k-1}\right\}=\frac{p_{ij}\times\mu^{i}(k-1\mid k-1)}{\sum_{i=1}^{r}p_{ij}\mu^{i}(k-1\mid k-1)} \tag{8-5}$$

2. Model update. For the filter matched to the model Mj(k), the update is given as following, take an example of non-linear system, the model update for a Gaussian filter is given in (8-1) and (8-2). Then the model-conditioned updated equations for the state estimate and the covariance are given as following

$$\hat{x}^{j}(k\mid k-1)=g^{j}(k\mid k-1)x^{0j}(k\mid k-1) \tag{8-6}$$

$$\Sigma^{j}(k\mid k-1)=g^{j}(k\mid k-1)\Sigma^{0j}(k\mid k-1)[g^{j}(k\mid k-1)]^{T}+Q^{j}(k-1) \tag{8-7}$$

The innovation vector and innovation covariance matrix are given as

$$\nu^{j}(k)=z(k)-h^{j}(k)\hat{x}^{j}(k\mid k-1)-\delta^{j}(k) \tag{8-8}$$

$$S^{j}(k)=h^{j}(k)\Sigma^{j}(k\mid k-1)h^{j}(k)^{T}+R^{j}(k) \tag{8-9}$$

The Kalman filter gain for model j is computed as

$$K^j(k) = \Sigma^j(k \mid k-1)[h^j(k)]^T S^j(k)^{-1} \tag{8-10}$$

Finally, the updated estimate of state and covariance

$$\hat{x}^j(k \mid k) = \hat{x}^j(k \mid k-1) + K^j(k)\nu^j(k) \tag{8-11}$$

$$\Sigma^j(k \mid k) = [I - K^j(k)h^j(k)]\Sigma^j(k \mid k-1) \tag{8-12}$$

3. Model likelihood of the filter matched to Mj(k) is defined by

$$\Lambda_k^j = \left|2\pi S(k)\right|^{1/2} \exp\left(-\tfrac{1}{2}(\nu_k^T S_k^{-1}\nu_k)\right) \tag{8-13}$$

Model regime probability update for model $M_j(k)$ is given as

$$\mu^j(k \mid k) = \frac{\mu^j(k \mid k-1)\Lambda^j(k)}{\sum_{n=1}^{r}\mu^n(k \mid k-1)\Lambda^n(k)}; \quad n \in r \ \forall \, all \ r \text{ models} \tag{8-14}$$

where $\mu^j(k \mid k-1) = \sum_{i=1}^{r} p_{ij}\mu^i(k-1 \mid k-1)$, $\mu^j \geq 0$; and $\sum_i^r \mu^j = 1$

4. The overall combination of state estimate and error covariance for IMM are

$$\hat{x}(k \mid k) = \sum_{j=1}^{r}\mu^j(k \mid k)\hat{x}^j(k \mid k) \tag{8-15}$$

$$\Sigma(k \mid k) = \sum_{j=1}^{r}\mu^j(k \mid k)\left\{\Sigma^j(k \mid k) + [\hat{x}^j(k \mid k) - \hat{x}(k \mid k)]^T[\hat{x}^j(k \mid k) - \hat{x}(k \mid k)]\right\}$$

8.2 Formation of model bank

The IMM bank consists of many models as shown in Fig.8-1. In dynamic SLAM, when the robot is traversing through a rather quiet and spacious environment, the state vector and its covariance can be modeled linearly and approximated by a linear estimator such as a Kalman filter. On the other hand, if the robot encounters an obstacle and tries to avoid it by making a turning at constant speed, it can be modeled by a non-linear estimator. If a robot suddenly encounters an object moving towards it, it then must be modeled by different estimator which encompasses the possible maneuvering motion like sudden acceleration to move to the right or to the left. Normally, the first (linear) model

can be combined with the second (non-linear) model to reduce complexity. The third model shall cover the sudden shift of the motion between the robot and head-on moving objects. This mode can be modeled by augmenting the state vector with a larger covariance noise. As such, these models are the basic elements that form our model bank. All element models must also take into account for inherent uncertainty in robot's sensors (proprioceptive or exteroceptive), by adding the white random noise. These mathematic models are briefly described as follows (The details of formulating the models are presented in the later section of simulation and implementation.):

1. First Model (linear motion with nearly constant velocity)

This model represents the relative motion between the robot platform and the landmarks when robot is moving along a straight line and on relatively smooth surface with little process noise. The state process is

$$x_k = F(k, x_{k-1}, u_k) + G(k)\varepsilon_{k-1} \tag{8-16}$$

where x(k) is the 2-D state variable defined as ; x, y are Cartesian coordinates of position and are the velocity along x-axis and y-axis respectively. ε_k ~N (0, Q(k)) is the white Gaussian process noise with zero mean and covariance, Q(k), and it is used to account for the small turbulence like slippage, uneven surface, and other motor/engine/sensor related uncertainty. The state transition matrices are

$$F(k, x_{k-1}, u_k) = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_{k-1} \; G(k) = \begin{bmatrix} \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{2}T^2 \\ T & 0 \\ 0 & T \end{bmatrix} \tag{8-17}$$

where T is the scan sampling interval between time step t_k and t_{k-1}.

The observation process: $z_k = h(k, x_k, m_k) + \delta_k$ (8-18)

where $h(k, x_k, m_k)$ is a non-linear state vector to measurement relationship matrix

$$h(k, x_k, m_k) = [x(k), y(k), \dot{r}(k)]^T \tag{8-19}$$

where $\dot{r}(k) = \dot{x}(k)\cos\theta + \dot{y}(k)\sin\theta$ and $\theta(k) = \tan^{-1}\left(\Delta y / \Delta x\right)$; $\Delta x = x(k) - x(k-1)$; $\Delta y = y(k) - y(k-1)$. The polar coordinate can be converted into Cartesian $\dot{x}, \dot{y}$ vectors.

The measurement noise vector is denoted by $\delta_k \sim N(0, R(k))$ which is white Gaussian noise with covariance R(k) to present the measurement uncertainty.

2. Second and third Models (coordinated turn with slight maneuver in low speed) These models represent the situation when the robot encounters with obstacles and makes an evasive turn to avoid collision (either right-turn or left-turn). The motion uncertainty comes from the change of relative speed (acceleration) between the robot and the slow moving obstacles, i.e., a walking person. Normally the robot can only makes a constant turn which is represented by a turn rate ω. As for the sudden acceleration, it can be modeled by a significant process noise, ε_{k-1}, with zero mean and a properly designed covariance Q, which is normally larger than Q in first model. The 2-D state vector is defined as

$$x = [x, y, \dot{x}, \dot{y}]^T \tag{8-20}$$

where ω is the turn rate of maneuver. However, the state process model is similar to Eq. (8-16) except that the state transition matrices have to include the turn rate of maneuver ω expressed as

$$F(k,x_{k-1},u_k)=\begin{bmatrix} 1 & 0 & \frac{\sin\omega T}{\omega} & -\frac{1-\cos\omega T}{\omega} \\ 0 & 1 & \frac{1-\cos\omega T}{\omega} & \frac{\sin\omega T}{\omega} \\ 0 & 0 & \cos\omega T & -\sin\omega T \\ 0 & 0 & \sin\omega T & \cos\omega T \end{bmatrix} x_{k-1}; G(k)=\begin{bmatrix} \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{2}T^2 \\ T & 0 \\ 0 & T \end{bmatrix} \quad (8\text{-}21)$$

The mode-conditioned turning rates ($\omega = \omega_2$ or ω_3) are given as

$$\omega_2(k)=\frac{a}{\sqrt{(\dot{x}+\dot{x}_r)^2+(\dot{y}+\dot{y}_r)^2}}\ ;\ \omega_3(k)=\frac{-a}{\sqrt{(\dot{x}+\dot{x}_r)^2+(\dot{y}+\dot{y}_r)^2}} \quad (8\text{-}22)$$

where $\omega_2(k)$ is right turn rate and $\omega_3(k)$ is left turn rate; $a>0$ is the target maneuver acceleration. $(\dot{x}+\dot{x}_r)$ and $(\dot{y}+\dot{y}_r)$ are Cartesian coordinates of relative velocity between the target and the robot. Since the observation measurement contains only the position information, so it can be modeled as

$$z(k)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x(k)+\delta(k) \quad (8\text{-}23)$$

3. Other Models (uncoordinated moving objects in place with more maneuver)

This kind of modes can be modeled as the second model except that the process noise and measurement uncertainty may be modeled by either a higher level of covariance or an auto-correlation colored noise instead of white Gaussian noise to cover the more severe maneuvering motion. In certain special scenario, the augmented state may be necessary to accommodate the higher order of non-linearity. However, we restrict our study in low-speed and low-maneuvering robotic applications, and the high-order non-linear modeling is beyond the scope of this paper. Therefore, it is not further investigated in this work.

8.3 Clutter sensitive switching mechanism for IMM models

The effective IMM system requires a dynamic switching scheme for applying the right model to match the mode of the dynamic target behavior. The mode switching (or mode jump) process is a Markov chain process and its transition probability is assumed to be a time-invariant discrete event denoted as

$$P_{ij} \cong P\{M_j(k) \mid M_i(k-1)\} \quad \forall\, i, j \in r \tag{8-24}$$

and this is the probability of switching from the previous model, $M_i(k-1)$ to the current model, $M_j(k)$. Normally, P_{ij} is an element in a $r \times r$ matrix $\boldsymbol{M}_{ij}$, where r is the total number of models in the model bank. P_{ij} is constrained by the following conditions:

$$P_{ij} \geq 0 \text{ and } \sum_{j}^{r} P_{ij} = 1 \tag{8-25}$$

However, P_{ij} is normally determined prior to implementation of IMM simulation, and based on the parameters obtained from certain scenario empirically. The model switching takes place if:

1. The tracking filter is initialized under the constant velocity model (model #1) and a maneuver (model #2 or model #3) is detected, then the model transition from $M_1(k-1)$ to $M_2(k)$ (or $M_3(k)$) will take place. This detection of maneuver on-set is based on the following condition:

   The maneuver of an extraneous moving object can be detected by monitoring the normalized innovation squared (NIS), or Mahalanobis distance of a tracking filter. If it exceeds a preset threshold $\zeta_{\max}$, then it is assumed that object (the tracked target) has deviated from its previous "pattern" [Shalom, Li, and Kirubarajan, 2001]. The

threshold is set up according to the pdf of $\zeta(k)$ which is Chi-square distributed so that the probability of the NIS below this threshold is given as

$$P\{\zeta(k) \le \zeta_{\max}\} = 1 - \varepsilon \tag{8-26}$$

where ε is the tail probability and chosen to be very small, say, 0.01. For our 3S-2D sliding window, we can replace the innovation ζ by a moving sum of the NIS over this sliding window of three scans such that

$$\zeta^s(k) = \sum_{j=k-s+1}^{k} \zeta(j) = \sum_{j=k-2}^{k} \zeta(j) \quad \text{for s =3} \tag{8-27}$$

The corresponding Chi-square distribution has ($L_w \times D_z$) degree of freedom, denoted as $\chi^2_{L_w, D_z}$. L_w is the time length of a sliding window (in our case, L_w =3); D_z is the dimension of the measurements (in our case, D_z =2). The above Chi-square distributed density function is given in Appendix B.

2. As the maneuver is over and we want to terminate the model #2 or model #3 and return to model #1. Again, when the fading average innovation falls below the threshold, the IMM switches back to $M_1(k)$ from $M_2(k-1)$ or $M_3(k-1)$. Note that the maneuver is the relative motion between the robot and the targets. For a stationary target (landmark), this motion is based only on robot's obstacle avoidance algorithm, and this algorithm will help determine the setting of model transition probability, P_{ij}.

A schematic illustration of IMM processing with two models in the model bank is shown in following figure 8-2.

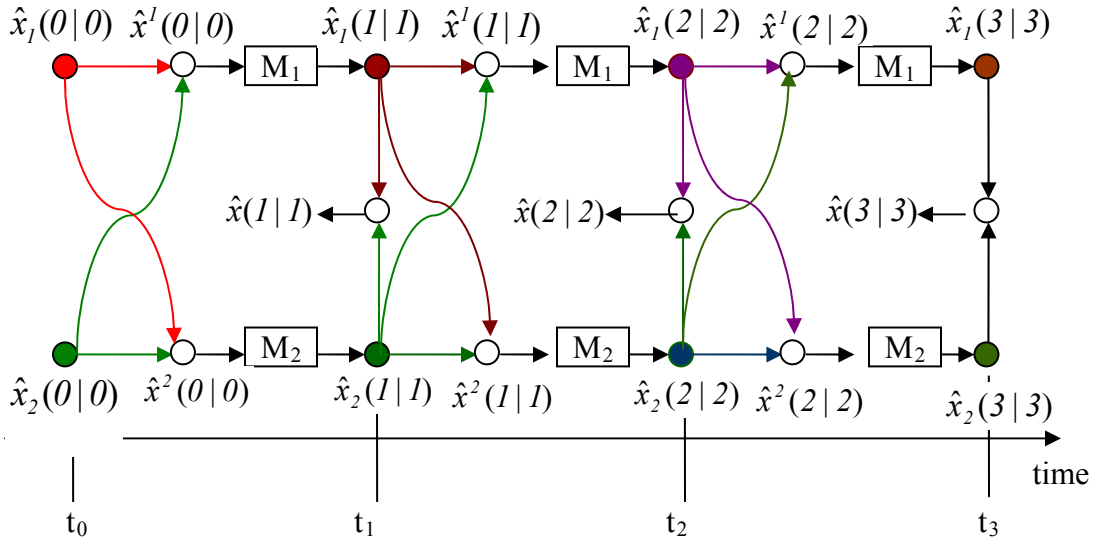


Figure 8-2. Signal flow diagram of IMM processing

8.4 IMM output for SLAM

The output of an IMM filter is a Gaussian mixture as in (8-15). The mixture of regime probabilities $\sum_i^r \mu^j = 1$ is analogous to there-sampling weighting in particle filter. Therefore, if more models with finer incremental parameters are included into an IMM's model bank (analogous to more particles used in the particle filter); the better output would be expected. In case we want to track the fast moving target, we then need to switch to the model with higher level of process noise, Q to accommodate for the observed deviation of tracking. However, if we just want to differentiate the moving object from static landmarks, we may just abandon the SLAM update and mark the observed measurement as "clutter" in the tentative track file. On the other hand, the output of state vector and the covariance may contain the “regime” model ID (model in effect) at time k for the purpose of target type categorization (i.e., to identify the moving object or stationary landmark) if necessary.

Chapter 9

Simulation and Results Analysis

In the chapter, we implement the proposed IMM-NNJPDA data association for SLAM. This algorithm is implemented in a simulated environment. In this work, we simulate both indoor and outdoor scenarios.

For the indoor environment, we assume that it is a warehouse-like building structured with many pillars, storage racks, and randomly placed stockpiles. The only passages are the aisle spaces between the array of these stationary objects (they may be regular or irregular in shape). Besides, there are some forklifts and workers occasionally moving around in the area. The robot travels through the aisles and makes turns if encountering the junctions of the aisles or obstacles. Along the way, it may encounter randomly generated "clutter" and moving objects. The robot is assumed to autonomously patrol the area through the unoccupied passages, and to build the map along the way.

For outdoor scenario, we simulate an environment by traversing along the peripheral sidewalks around a campus. For both scenarios, the targets (Landmarks) are also randomly generated initially. But they remain the same for each successive run during simulation process.

9.1 Simulated robotic platform and sensor

The robot platform to be simulated is a wheeled mobile robot (i.e., Mobilerobots^{TM} Pioneer series robot shown in Figure 9-1). The landmark feature measurements are assumed to come from a single scanning range measurement sensor (i.e., SICK^{TM}

Figure 9-1. Pioneer Robot platform with SICK laser scanner.

LMS200 laser scanner). The following are the parameters for simulation:

1. Robot max. speed: 0.7 m/s (based on Pioneer 3AT)
2. Speed encoder error: 0.5 m/s (on flat floor)
3. Turning angle error: 0.05 rad/s
4. Max. Measurement range: 10 m (based on LMS200)
5. Measurement range error: 0.1 m
6. Measurement bearing error: 0.5º
7. Spurious clutter spatial density: uniformly distributed.
8. Spurious measurement pdf (noise): Poisson distributed.
9. Max. speed of slow moving object (based on average human): 5km/h = 1.4 m/s
10. Sampling time interval: 0.1 s
11. Scenario area of simulation: 10 x 10 m^2

Before we start the simulation, we have to define the criteria of performance evaluation, and then compare our proposed algorithm with other algorithms in terms of these measures. All the comparison will be based on a set of Monte-Carlo simulations.

9.2 Performance and Evaluation metrics

There are two major metrics to evaluate the performance of the SLAM algorithms. First one is the position error along the trajectory; the other one is how soon all the landmarks on the map can converge to their lowest bound of covariance. For data association filters, the metrics are computational complexity, number of data association hypothesis generated, polynomial time for feasible solution and number of unsolved pair of data association, filter consistency...etc. [Daum, 1996]. Normally, the RMS error (as well as

velocity error) is used instead of the position error for quick glance of performance and it can be computed as

$$RMS(k) = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(x_i^k - \hat{x}_i^k)^2 + (y_i^k - \hat{y}_i^k)^2} \tag{9-1}$$

where $(x_i^k - \hat{x}_i^k)$ and $(y_i^k - \hat{y}_i^k)$ denote the true and estimated landmark x-y position errors at time k and i^{th} Monte-Carlo run; M is the number of independent Monte-Carlo simulation runs.

The other important metric is the filter stability. This involves the convergence of the covariance to a finite steady state – that is, whether the error is bounded or not. For the dynamic systems, which are normally unstable, the only requirement for stability of the filters (estimator) is the observability which depends on the initial covariance and therefore does not satisfy the "BIBO" (bounded input-bounded output) condition of stability. In view of this, we adopt another metric – the consistency of the filter, which can be defined as convergence of the estimate to the true value. In other words, with the increasing amount of information received, the uncertainty of a parameter should asymptotically reduce to zero within a finite time. One of the criterion we can use is the average of the normalized squared estimation error (NSEE), which must be equal to the dimension of vector if it is chi-squared distributed such that

$$E\left[(x_i^k - \hat{x}_i^k)(S_i^k)^{-1}(x_i^k - \hat{x}_i^k)^T\right] == \varepsilon(k) \le n_x \tag{9-2}$$

where the left side of equation is exactly the Mahalanobis distance or NIS (normalized innovation squared) used in calculation of the measurement likelihood as we described earlier in Chapter 5, and n_x is the degree of freedom. Therefore, the consistency of a filter should have an innovation of zero mean and white Gaussian statistic sense.

9.3 Setting up for simulation

Our simulation used to verify the validity of algorithms is based on three set of software code of Matlab environment. The first one is DynaEst™ 3.35 [Shalom, Y.B. 2001], a Matlab-based filter and estimator. It provides an easy and user-friendly interactive GUI (design graphic user interface) environment to implement the algorithm simulation and visualization of the results in terms of statistic performance evaluations. The procedure of simulation in DynaEst is shown as follows:

1. Create a project from the interactive window within the Matlab environment.
2. Select the data sources from external files or create the simulated sources within DynaEst. These data sources include the ground truth, path of robot generated from other sources, or the system modeling parameters from outside sources.
3. We can build our own motion or measurement equations that have been used to model our algorithms by selecting or typing the specific state variable matrix or vectors such as dynamic motion matrix F(t) or G(t), process noise covariances, Q(t), measurement function matrix H(t), and measurement noise, R(t) ...etc.
4. Besides, we can select the motion models, linear or non-linear with constant velocity or maneuvering acceleration, mode probability, and probability of transition between models. They can be customized to our need in order to formulate our algorithms.
5. The number of Monte Carlo run can be set to suit our requirement for statistic samples. The length of time is based on k time step scale, called revisit number.
6. The result of simulation can be plotted as figure files in Matlab environment or exported as data files to external storage for future use.

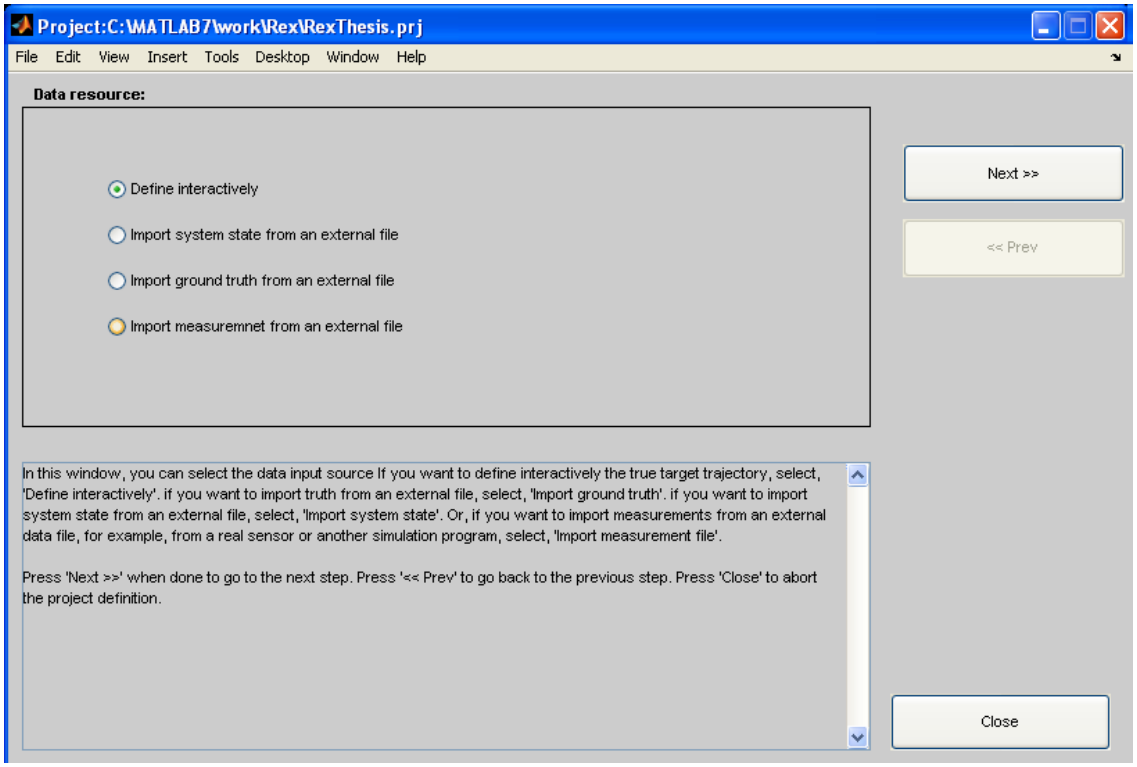


Figure 9-2. Create a new project in DynaEst environment for simulation

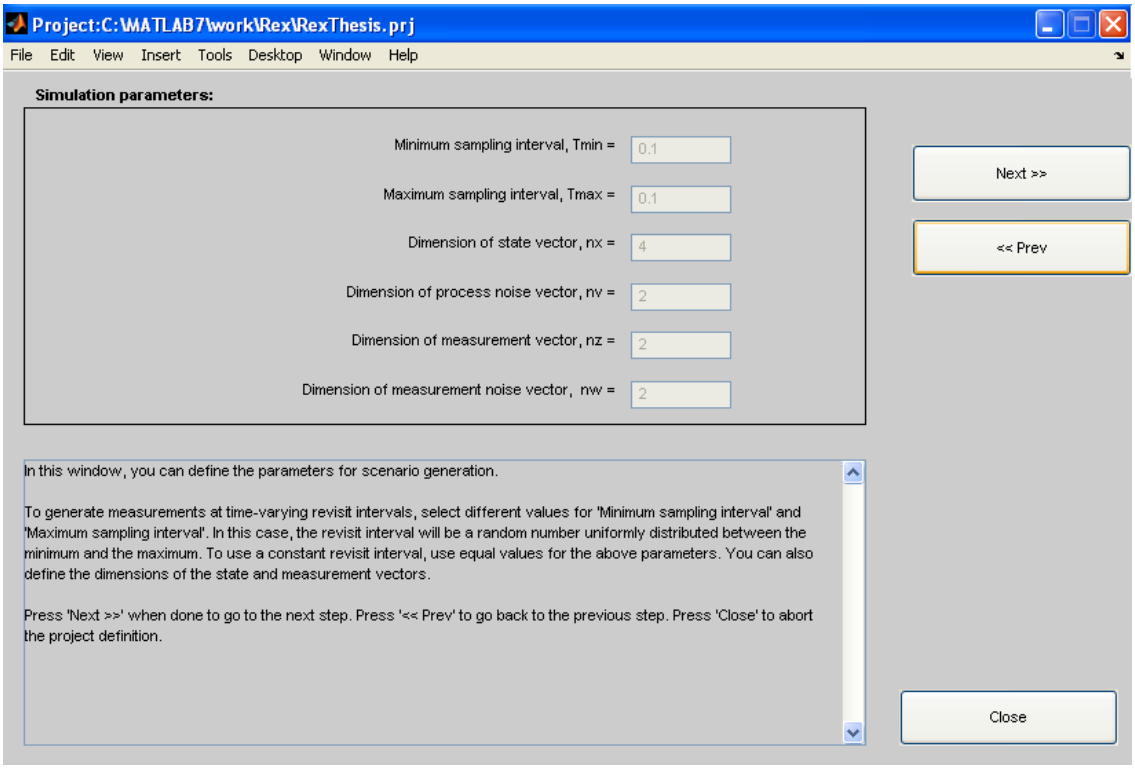


Figure 9-3. Window pane for selection of parameters.

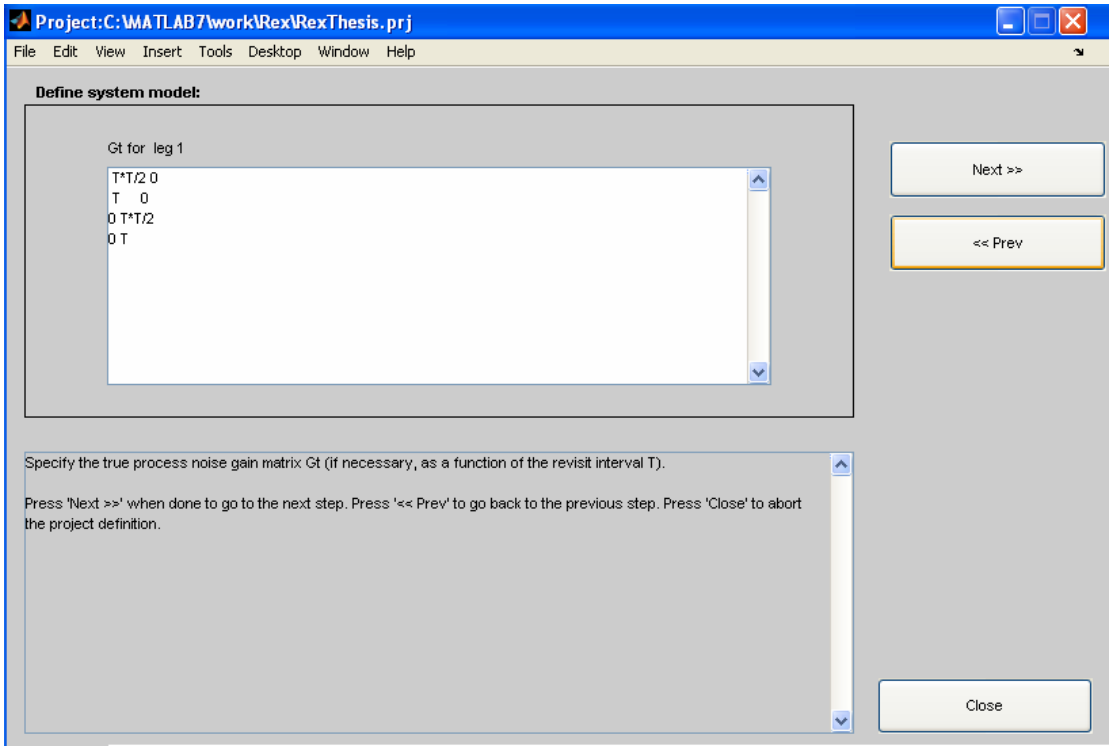


Figure 9-4. Design of our linear motion process matrix for model 1

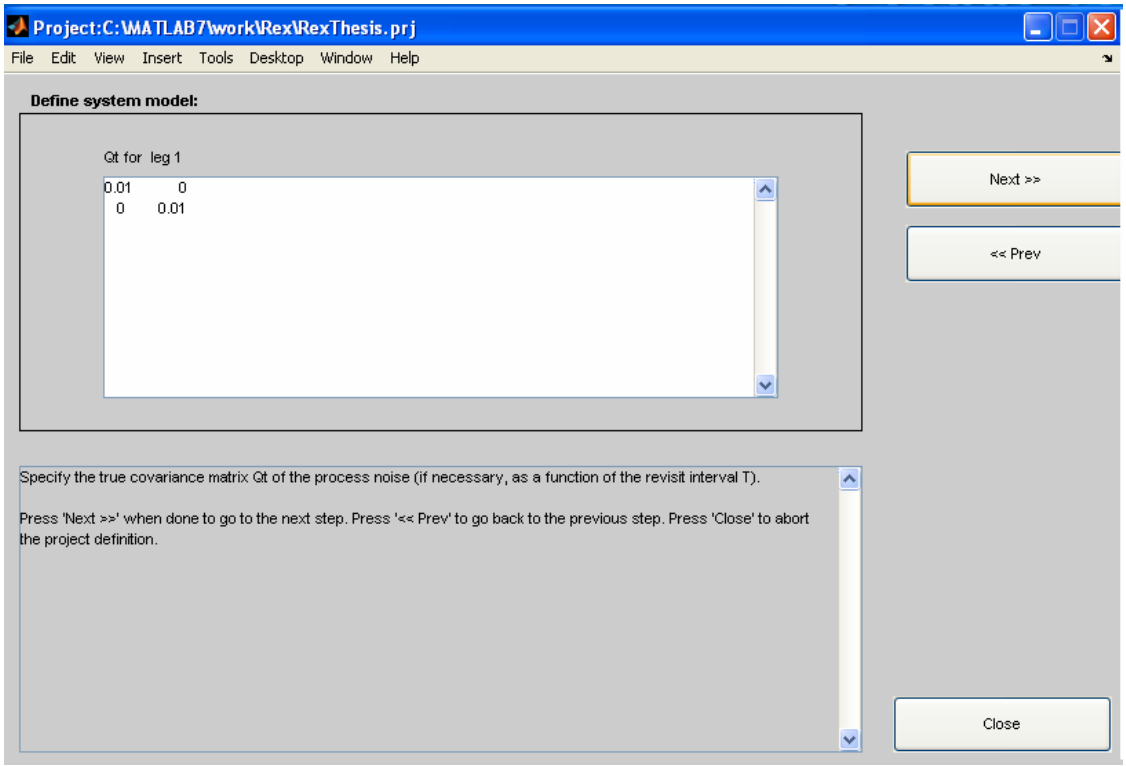


Figure 9-5. Design of the process noise covariance matrix, Q(t)

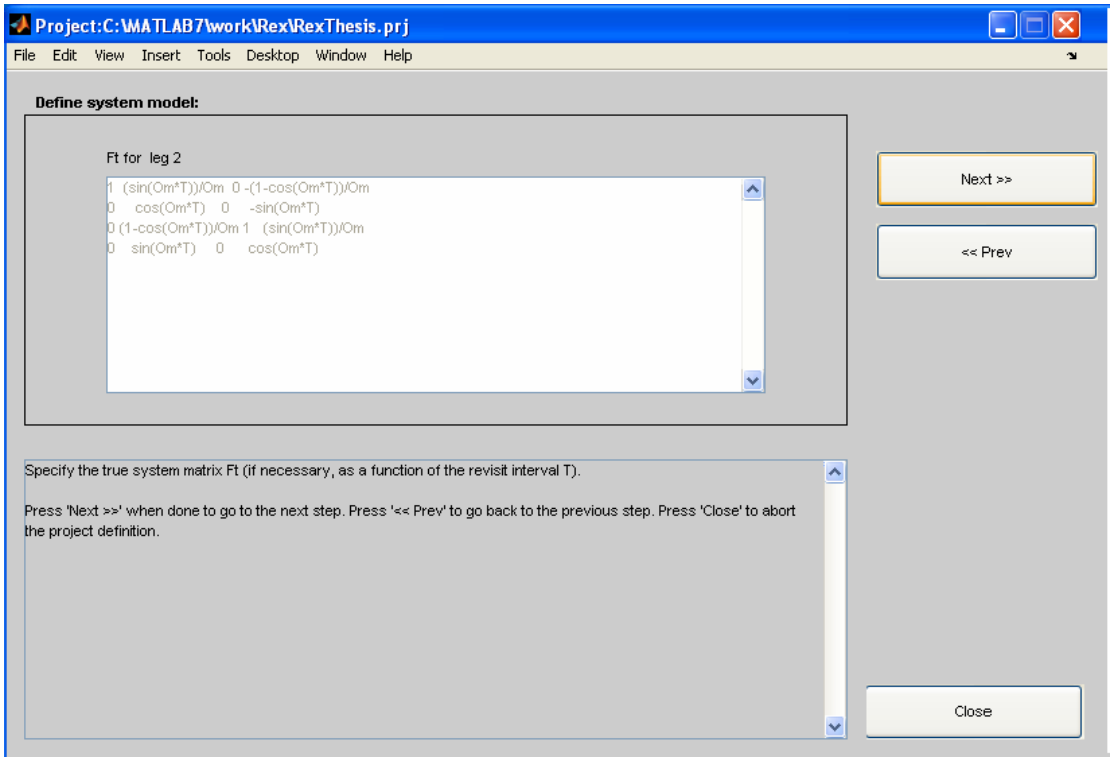


Figure 9-6. Design of nonlinear motion process matrix for model 2.

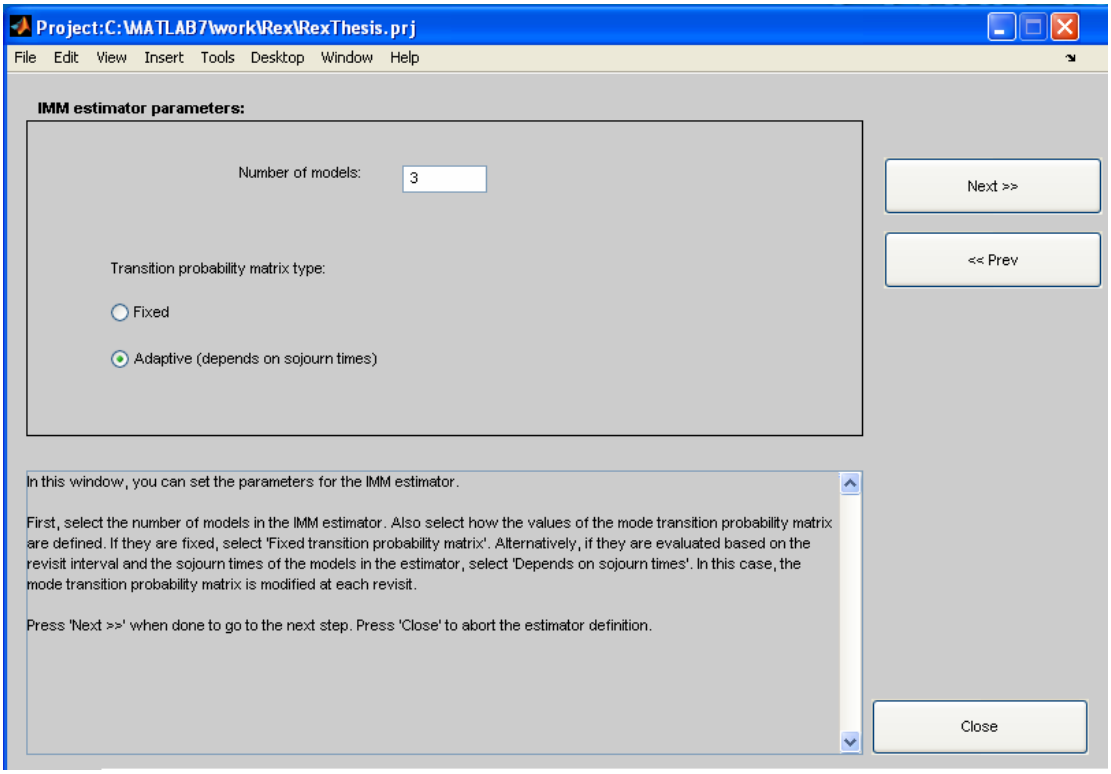


Figure 9-7. Choices of type of IMM systems

The second software we used for simulation is based on the Matlab Toolbox of EKF/UKF[1] which is provided by Jouni Hartikainen and Simo Sarkka from Helsinki University of Technology [Hartikainen, J, 2011]. It is very useful in generating the data and plots for performance comparison in terms of MSE or RMS errors. We use this program mostly for verifying the effectiveness of data association in SLAM. Therefore, the result is focused on the more macroscopic level of performance, such as the position uncertainty, time needed to convergence. The third software we used is based on the existing optimization linear programming algorithms. We tailored these programs to suit our asymmetric assignment algorithm for generating the optimal or sub-optimal solutions of one-to-one measurement-landmark association for JPDA. The performance metric is based on the number of successful pairing under the constrained conditions, the number of detections, and the number of miss or loss of tracks ...etc. This program code is listed in Appendix D.

9.4 Implementation and Result analysis

In order to provide the metrics of performance comparison, several times of simulation with Monte Carlo method are run for each of the following data association algorithms:

1. GNN with one-to-one assignment (single model – 1scans)
2. GNN with one-to-one assignment – IMM (multiple model scheme – 1 scans)
3. NNJPDA with 3S-Q Best assignment (single model scheme – 3 scans)
4. NNJPDA with 3S-Q Best assignment – IMM (multiple model scheme – 3 scans)

We first use an indoor warehouse as background scenario for this simulation. The environment is filled with stack of goods and the racks and some of them may serve as

[1] Matlab Toolbox EKF/UKF is provided by J. Hartikainen at http://www.lce.hut.fi/research/mm/ekfukf/

features of landmarks, shown as red circles in Figure 9-8 (The moving objects is shown as the red double dots at the end of simulation.). The blue diamonds are measurements from the simulated onboard sensor.

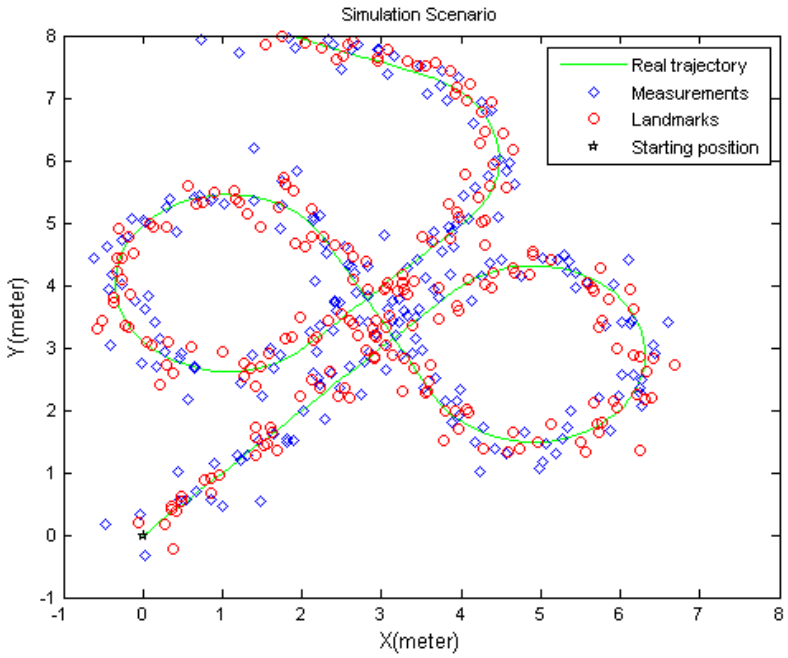


Figure 9-8. The simulated dense environment

The model parameters for the tracking (data association) filters are set as follows:

1. Covariances of process noise for Model 1, Model 2, and Model 3:

$$Q_1 = \begin{bmatrix} q_{11} & 0 \\ 0 & q_{22} \end{bmatrix} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}; \; Q_{2,3} = \begin{bmatrix} q_{11} & 0 \\ 0 & q_{22} \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \tag{9-3}$$

2. Covariances of measurement noise for the sensor:

$$R = \begin{bmatrix} r_{11} & 0 \\ 0 & r_{22} \end{bmatrix} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \tag{9-4}$$

3. The transition probability between models is set as:

$$p_{ij} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.9 & 1 & 0.1 \\ 0.9 & 0.1 & 1 \end{bmatrix} \tag{9-5}$$

9.4.1 Indoor performance evaluation

The robot trajectory of different data association filters is shown in Figure 9-9.

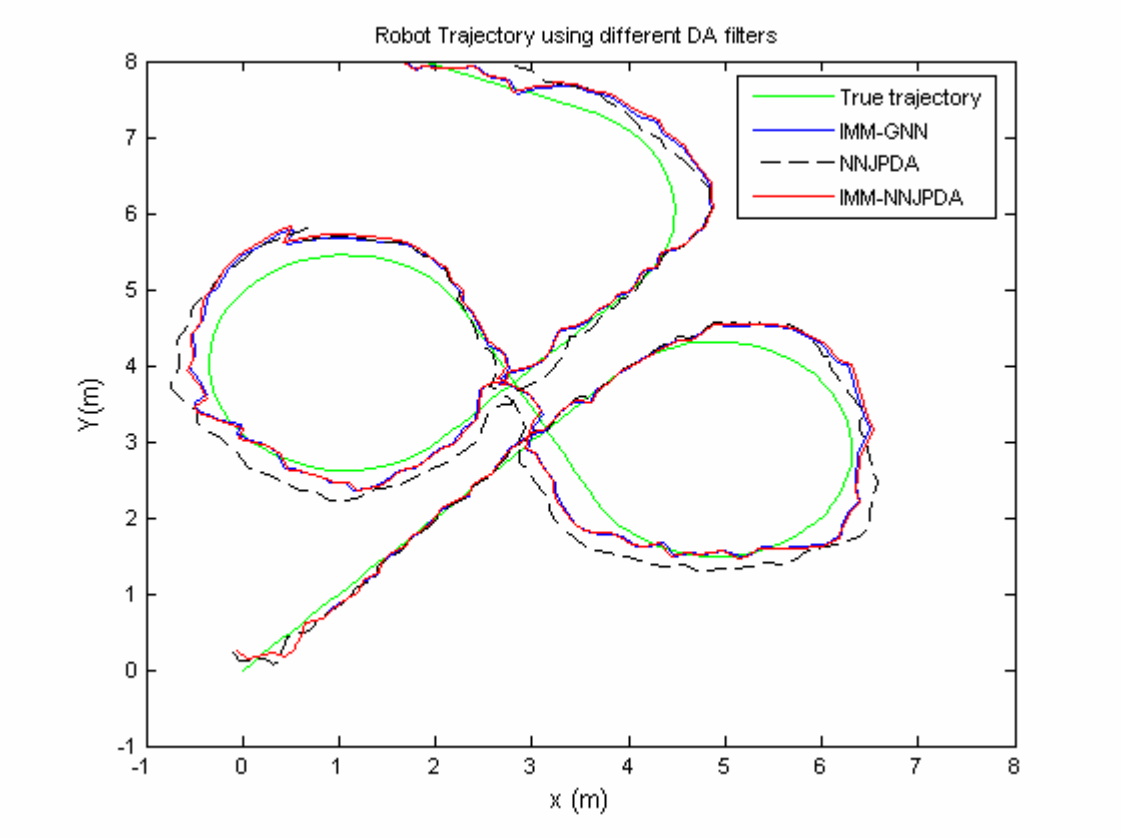


Figure 9-9. The trajectories using different filters

The robot has traveled from the start point (the origin) to its destination via the pathway (aisles between the stockpiles). The journey is broken down into several sojourns as shown in Figure 9-9. It begins its journey by going along a straight aisle and this first sojourn ends in 5 seconds. And then it made a right turn for 4 seconds due to the obstacles. At the end of this sojourn, it resumed the straight-line motion with constant velocity for 2 seconds until it encounters another obstacle which forces the robot to make turn again. This time it made a left turn and it took 4 seconds to come back to the previous crossing point. Once it reached this point, it resumed a straight line traveling again for 2 seconds. It then made the third turn (left) to avoid the obstacle ahead for the

rest of sojourn. The entire journey took 22 seconds. The sampling interval between each scan is 0.1 second, and therefore they are 220 time-steps during this process.

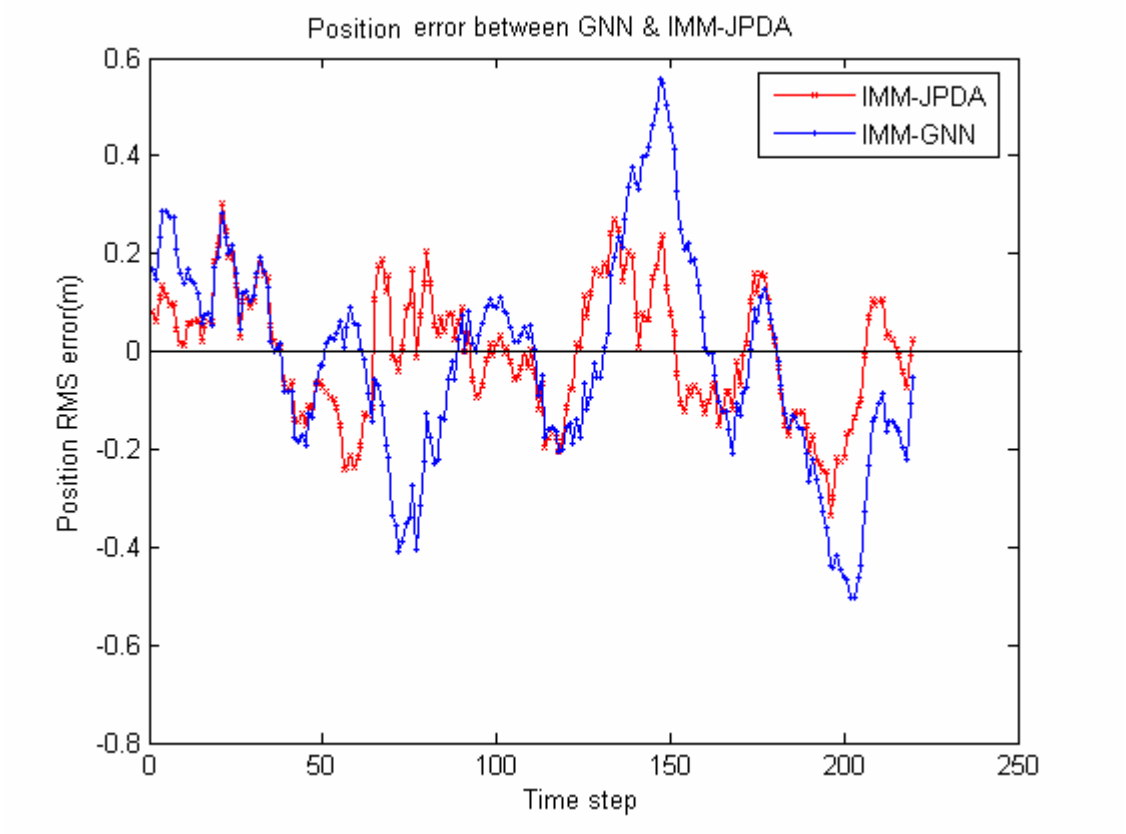


Figure 9-10. Data association RMS position error

We mainly compare our proposed data association algorithm with GNN (global nearest neighbor) which is considered as paradigm data association of superior performance according to the survey done by H. Leung [Leung, 1999] for the sake of plotting clarity. We first compare their performance in tracking the position of the robot (localization) in Figure 9-10. It shows our IMM data association filter outperforms GNN during the sojourn of a moving object which maneuvers to avoid collision (between 4-5 seconds and 12-15 seconds).

The average innovation for state update is also in favor of the IMM-JPDA as shown in Figure 9-11. It is due to the effect of one-to-one assignment algorithm of JPDA

with the delay decision of sliding window which greatly reduces the uncertainty in landmark and measurement association.

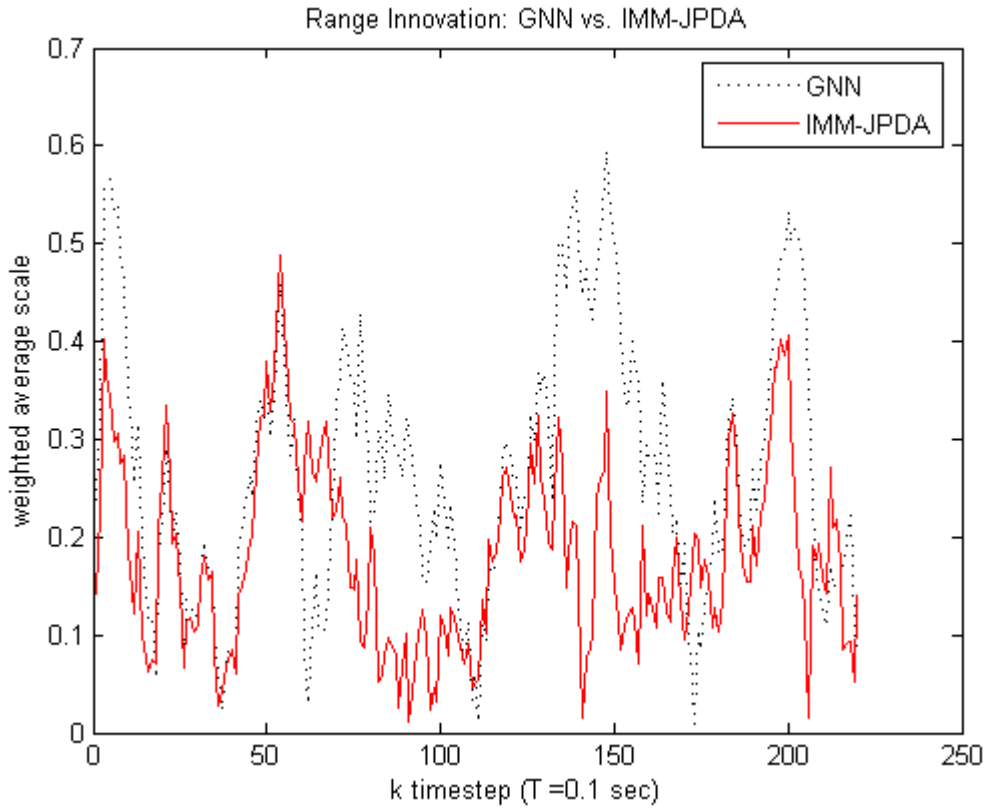


Figure 9-11. Innovation of data association filters

Another important metric to determine the filter efficiency is the filter consistency which is defined as follows:

Lemma 9-1: An estimator of a random parameter is said to be a consistent estimator if the estimate which is the random variable converges to the true value in some stochastic sense. Using the convergence in mean square error criterion, it requires that

$$\lim_{k\to\infty} E\left\{\left[\hat{x}(k,Z^k)-x\right]^2\right\}=0 \tag{9-6}$$

where the expectation is over x and Z^k. [Shalom, 2001].

The filter consistency in term of NIS error shown in Figure 9-12 indicates the accuracy of the tracking filters to estimate the target modes except at the time of 12 seconds and 15 seconds. We assume this may be due to the modeling error in disturbance (noise covariances) or the error of approximation which would affect the convergence of the filter. For example, we select $(\sigma_{11})^2 = 5\text{m/s}^2$ and $(\sigma_{22})^2 = 20\text{m/s}^2$ as parameters in the diagonal matrix of Q, which is supposed to accommodate the maneuver of moving objects. This assumption may be too big for a low-speed walking people or forklifts around. In addition, the noise covariance R of the measurement also needs fine-tuning if different kind of sensor is used.

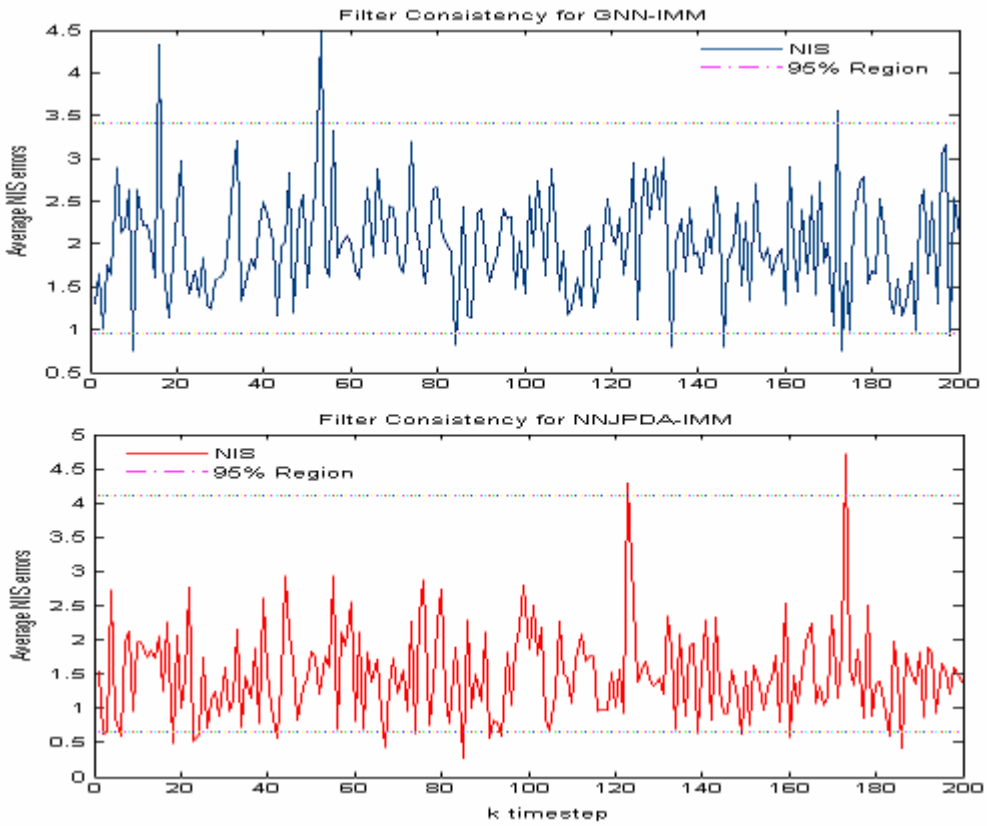


Figure 9-12 Filter consistency

The effect of changing covariance on the position uncertainty can be manifested by the asymptotically decreasing size of ellipses of landmarks as the robot traverses and

re-visits the landmarks along the path. This proves the correct data association takes toll in the SLAM scenario as shown in the following snap-shots of SLAM animation.

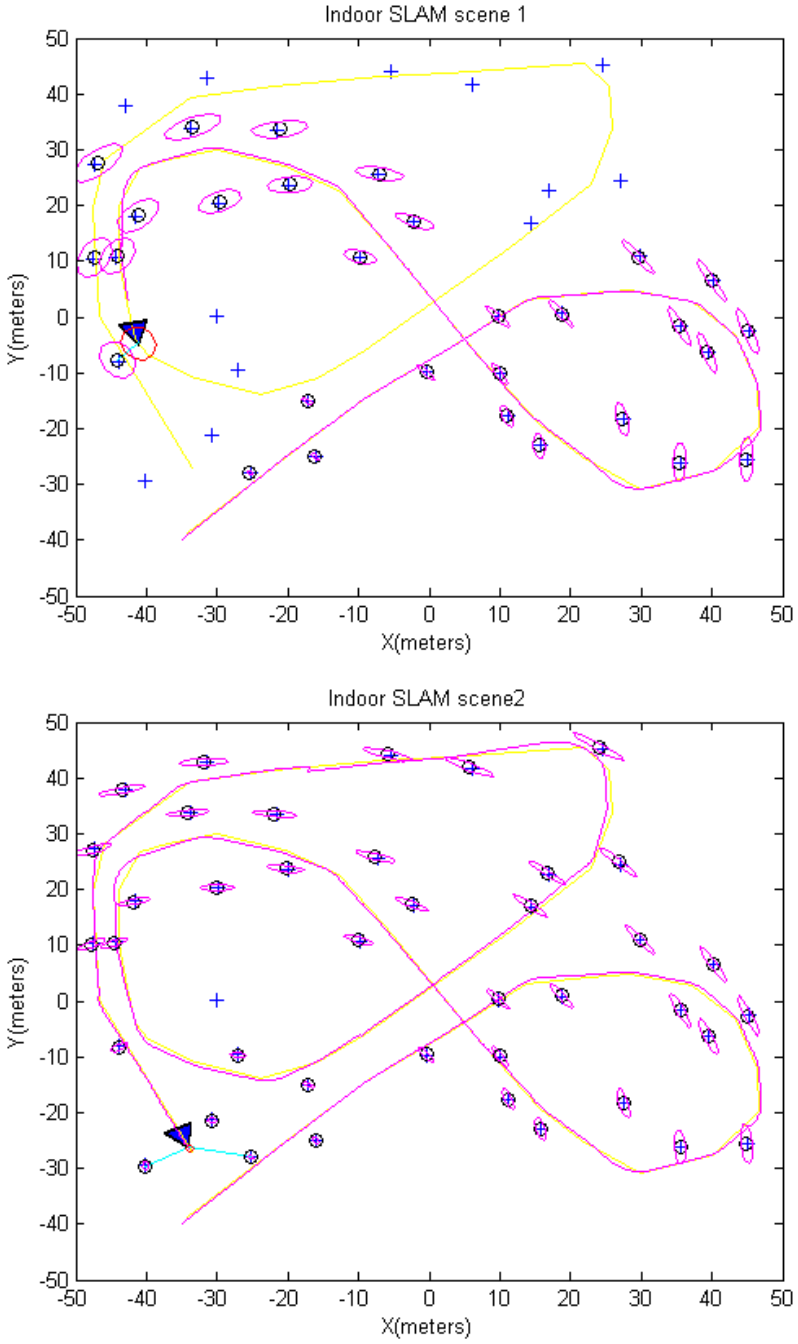


Figure 9-13. The reduction of position uncertainty results from correct data association.

The other way to show the decreasing covariance is the numerical data generated during the above simulation such as the landmark state covariance at different time step.

```
>> ST(2000).PV
ans =
    0.3337   -0.1553    0.0087
   -0.1553    1.0372    0.0056
    0.0087    0.0056    0.0014
>> det(ST(2000).PV)
ans =
  3.4055e-004
```

Figure 9-14 (a) State covariance at k=2000

```
>> ST(3000).PV
ans =
    1.7417    0.5118   -0.0349
    0.5118    0.4993   -0.0168
   -0.0349   -0.0168    0.0009
>> det(ST(3000).PV)
ans =
  7.1620e-005
```

Figure 9-14 (b) State covariance at k=3000

```
>> ST(5000).PV
ans =
    2.2955   -1.2485   -0.0386
   -1.2485    0.7951    0.0177
   -0.0386    0.0177    0.0009
>> det(ST(5000).PV)
ans =
  4.5149e-005
```

Figure 9-14 (c) State covariance at k=5000

```
>> ST(6000).PV
ans =
    1.0058    0.3821   -0.0174
    0.3821    0.2445   -0.0080
   -0.0174   -0.0080    0.0004
>> det(ST(6000).PV)
ans =
  8.3155e-006
```

Figure 9-14 (d) State covariance at k=6000

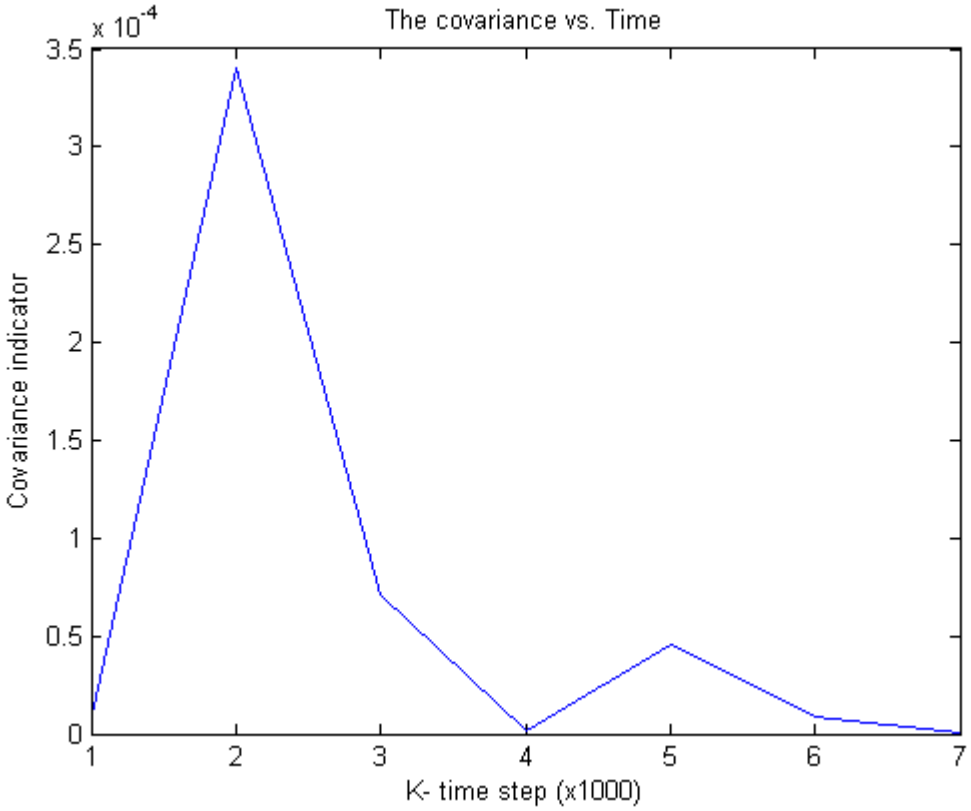


Figure 9-15. Asymptotically decreasing covariance

For the multiple model system, the performance of tracking depends on the proper switching between models. This switching mechanism depends on two factors. First is the model's regime probability. As shown in Figure 9-16, the probability of model 2 to take effect is as high as 0.8 when the right turn is about to be made at k = 40 (4 sec).

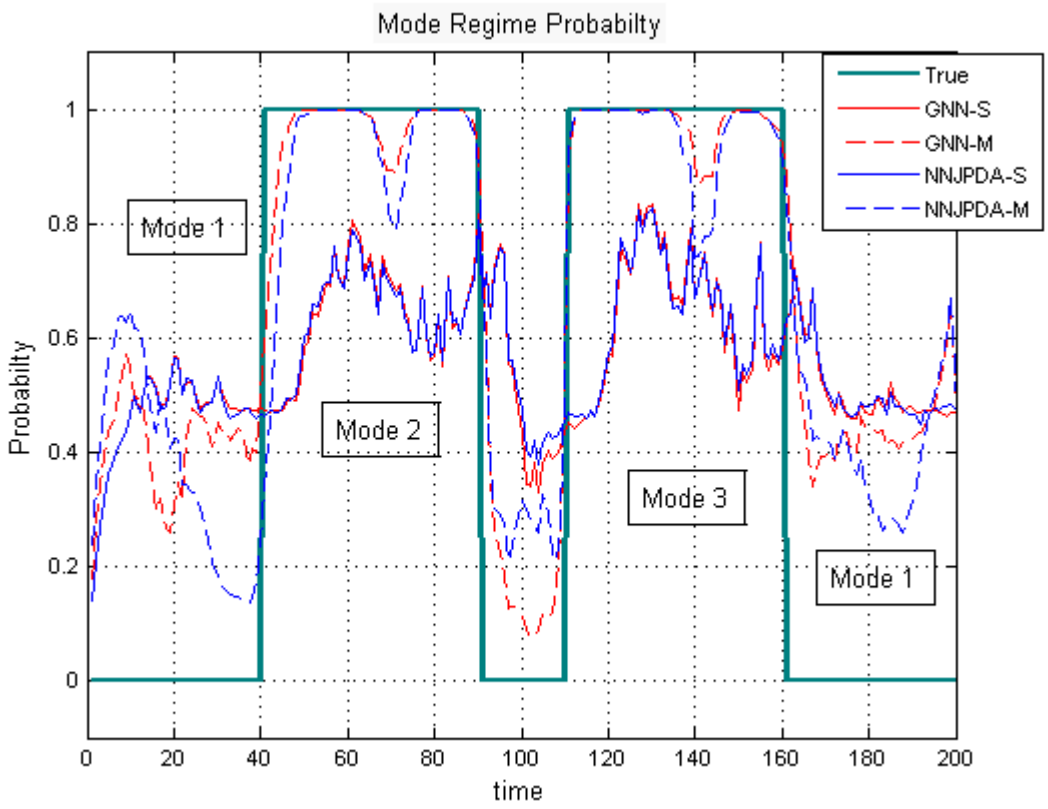


Figure 9-16. Probability of the model in effect

The second important factor is the transition probability P_{ij} which is shown in Figure 9-17. We assume the transition probability of switching from model 2 to model 1 is very easy (i.e., P_{21}= 0.9), since model 2 is the non-linear turning model; while model 1 is the constant speed and linear model. On the other hand, transition between two turning model (right turn and left turn) is less possible (i.e., P_{23} and P_{32} are only 0.1).

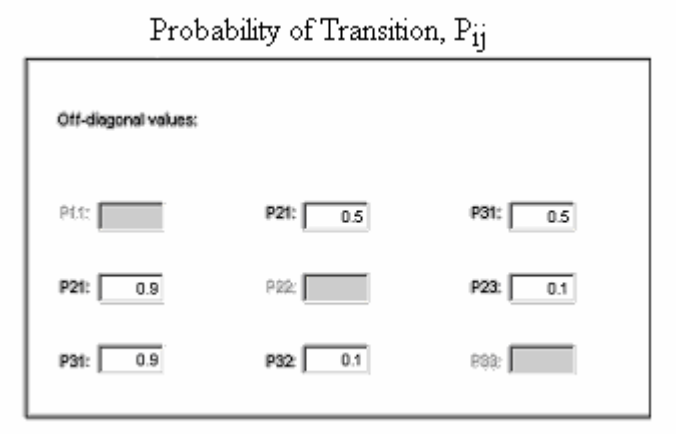


Figure 9-17. Model transition probability matrix

On the other hand, the filter gains obtained during the switching between models 1 and 2 is low and unstable due to maneuvering (as red curve in Figure 9-18). Conversely those gains (in blue) are relatively high and stable.

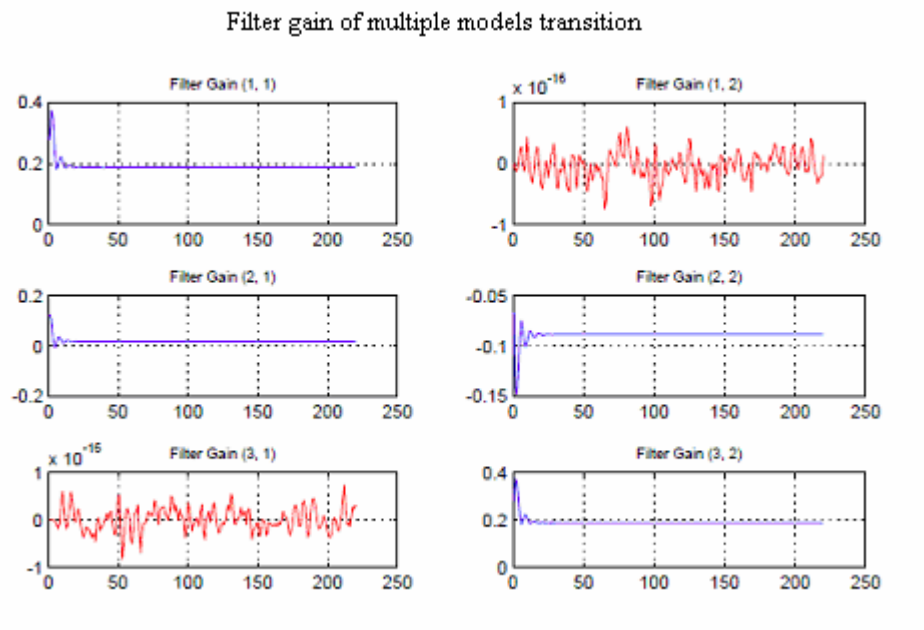


Figure 9-18. Filter gain affected by model transitions

When a moving object is approaching the robot, the evasive action will take place between the robot and the object (assume to be a person). The tracking algorithm shall adjust its estimator to activate the model which gets the proper parameters to

accommodate the change of bearing angle and the rate of turning. This ability is manifested by the filter's response to the onset situation. Figure 9-19 shows the sensitivity of our proposed data association filter which respond to the maneuvering turns.

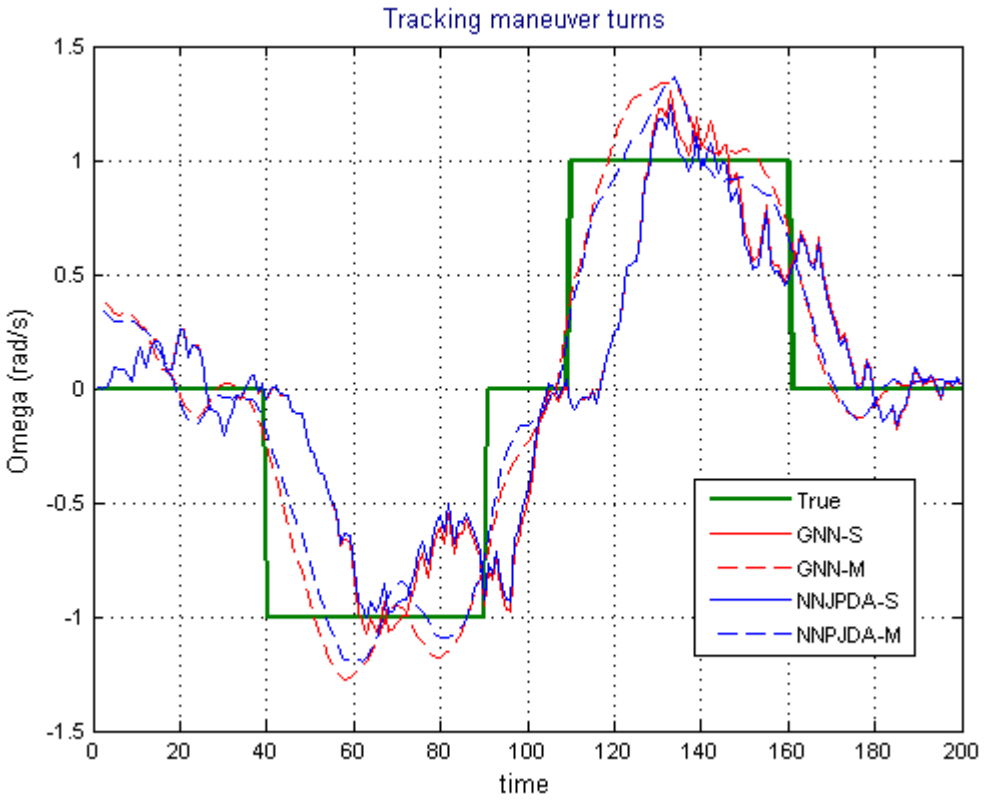


Figure 9-19. Tracking scenario of maneuver turns

An average RMS error comparison is listed as the output of computer simulation:

LAB7\work\Rex

Command Window

```
Average Root Mean square errors of position estimates:
GNN-S = 0.1387
NNJPDA-S = 0.0625
GNN-IMM = 0.1318
NNJPDA-IMM = 0.0675
Warning: Ignoring extra legend entries.
> In legend at 239
```

Figure 9-20 The average RMS error for DA filter in localization

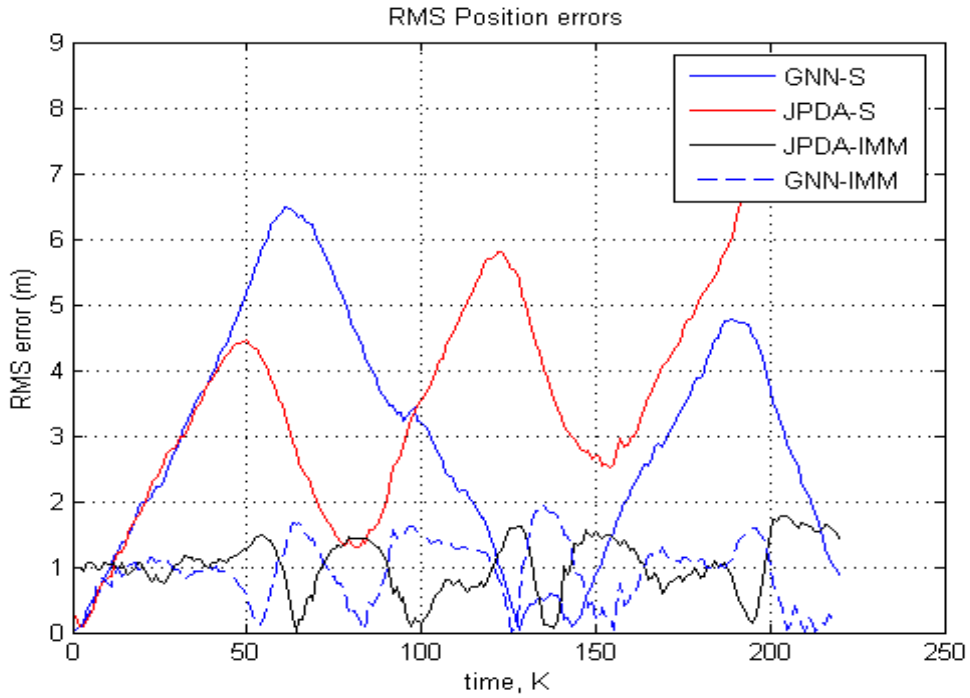


Figure 9-21 RMS position errors comparison

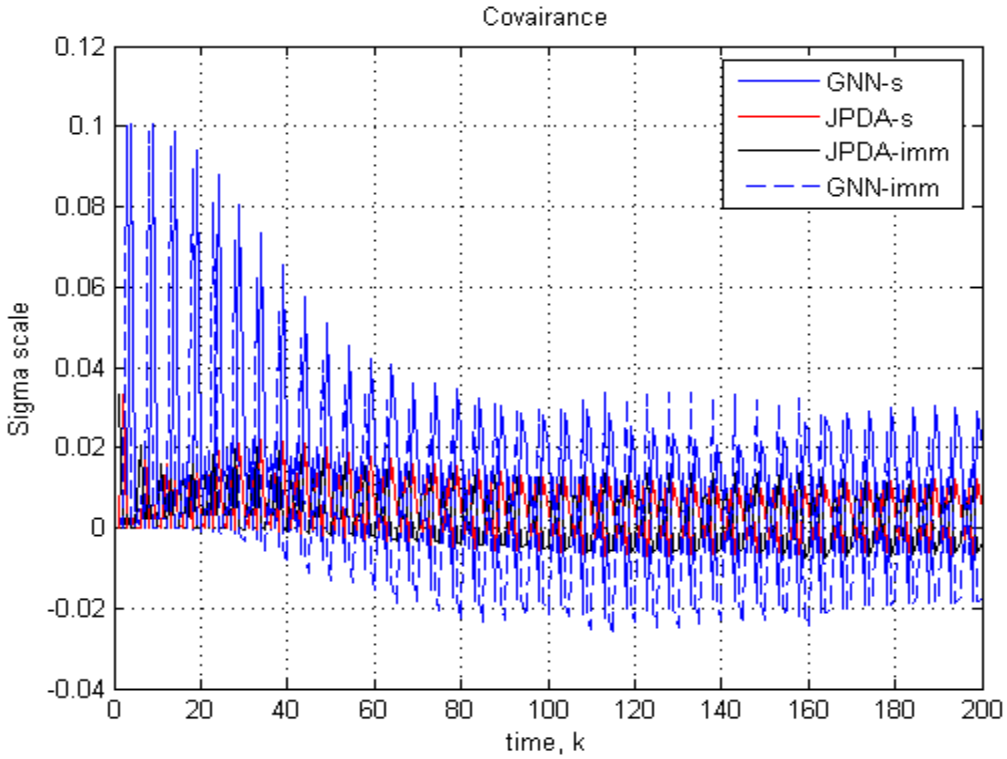


Figure 9-22 Average covariance size comparison at each time step

The real-time tracking scenario is simulated as viewed from laser scanner. The following figures are snap-shots of different time steps during running the simulation. Figure 9-23 represents the laser scanner-centric local map at k = 5 time step, where the landmarks are few and sparse so that they can be associated rather easily with the most probable measurement within the vicinity. The green lines stand for the tracking links of the landmarks and corresponding measurements. The loose blue dots are unassigned measurements that may be due to the clutter or new targets. However, as the robot moves along, more and more new landmarks are initialized and the more assignments between landmarks and measurements are confirmed and the pairs are successfully associated as shown in Figures 9-24 through 9-27.

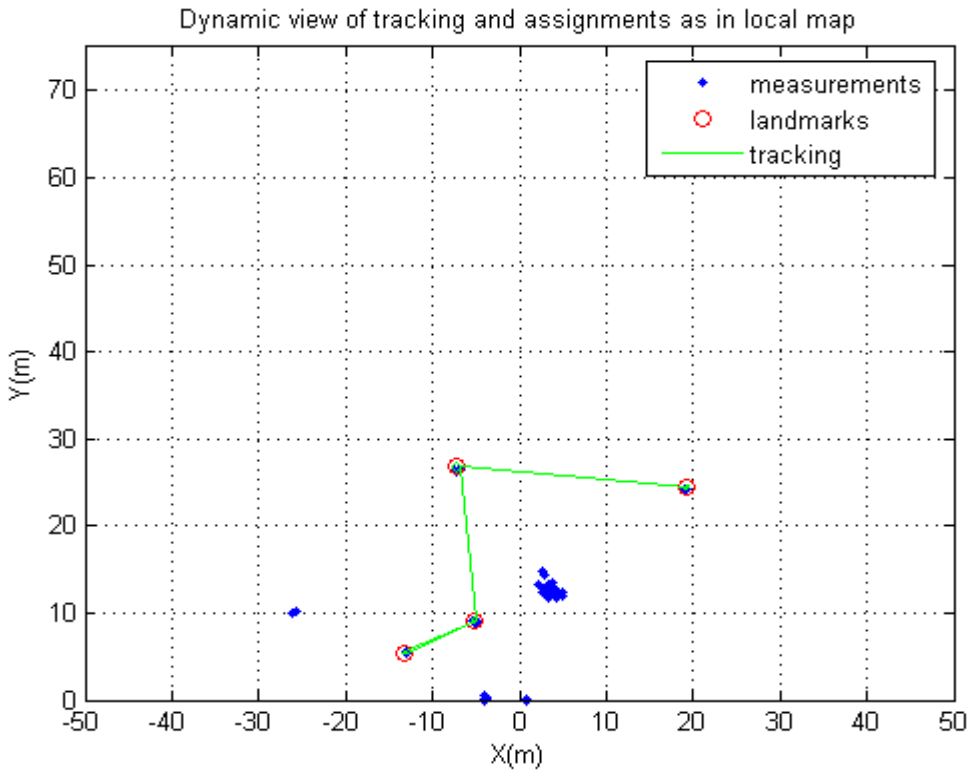


Figure 9-23 Tracking Data Association scenario at scan k= 5

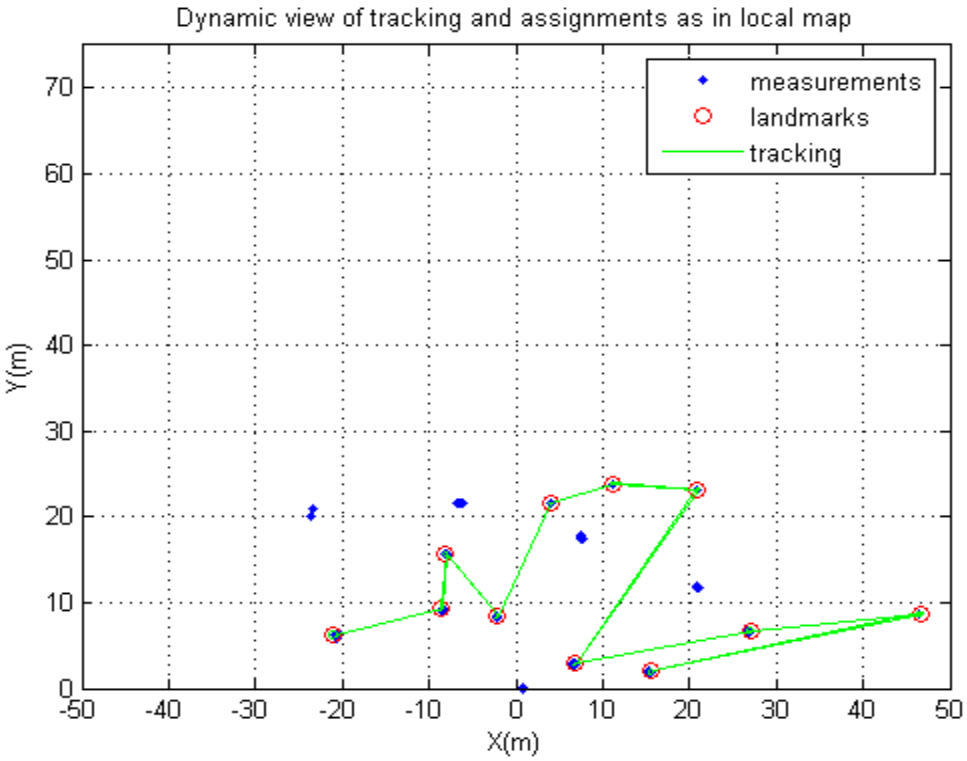


Figure 9-24 Tracking Data Association scenario at scan k= 8

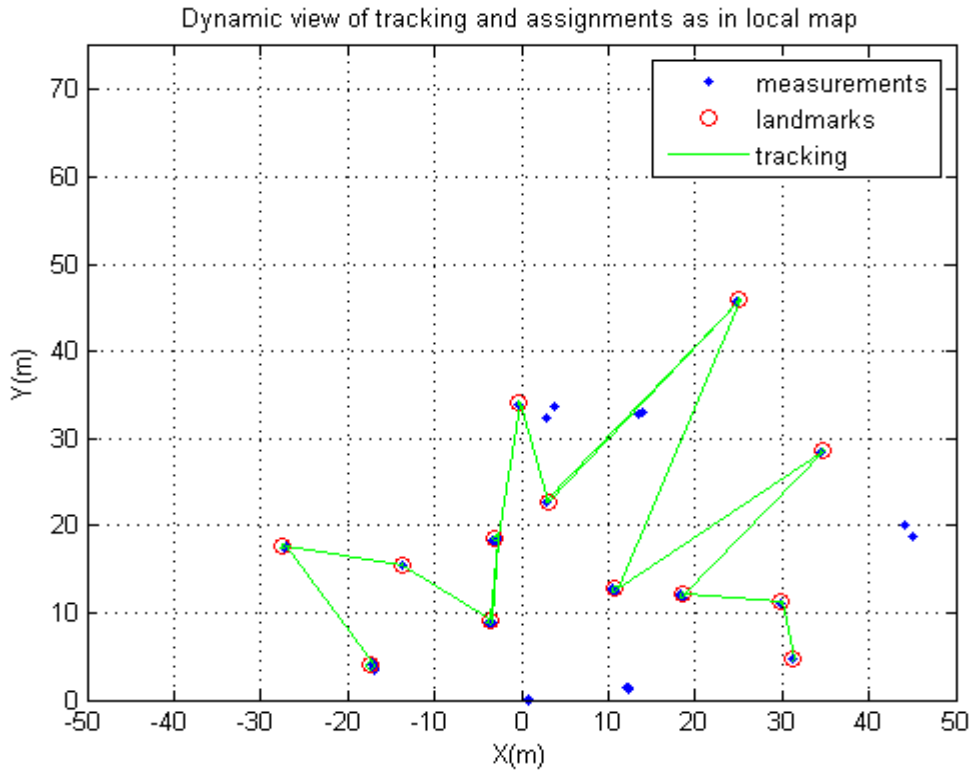


Figure 9-25 Tracking Data Association scenario at scan k= 13

However, as the robot starts to make turn due to the obstacles, the IMM model kicks in and maneuvering response causes the tracking filter error which manifests in track loss or re-initilaize the data association process. The number of tracking links become minimum (in Figure 9-23, there is only one link left.) and more measurements (blue dots) are left unassigned.

At time step k = 20, the robot has completed a circle and revisit many landmarks, therefore it can make a good trace of links and reduces the unassigned measuremnts to the minimum numbers as shown in Figure 9-25.

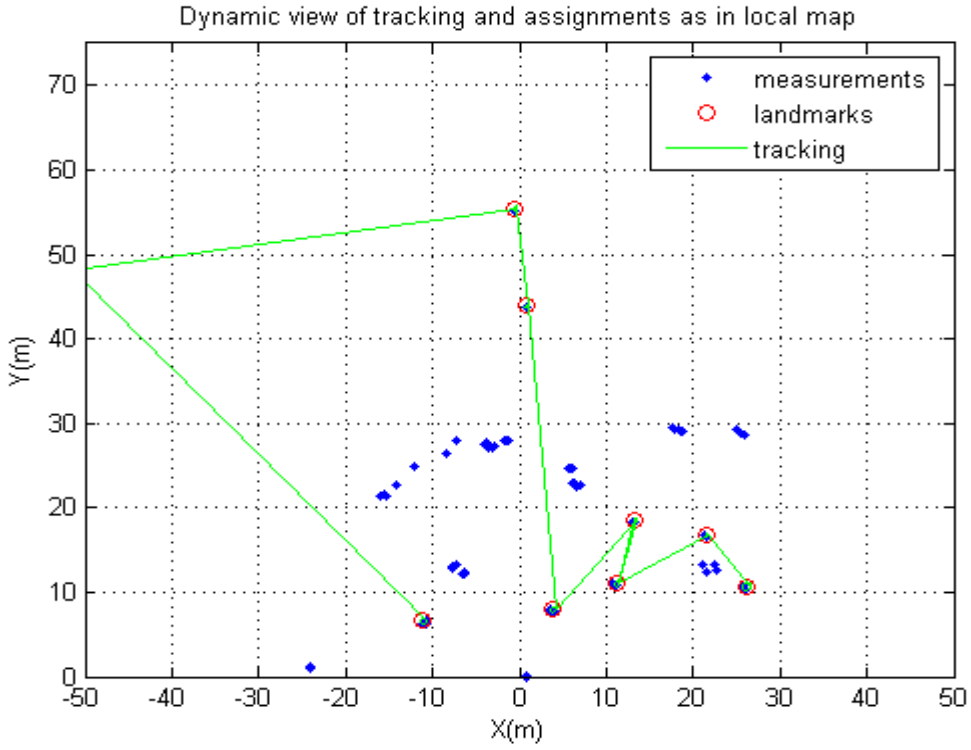


Figure 9-26 Tracking Data Association scenario at scan k= 16

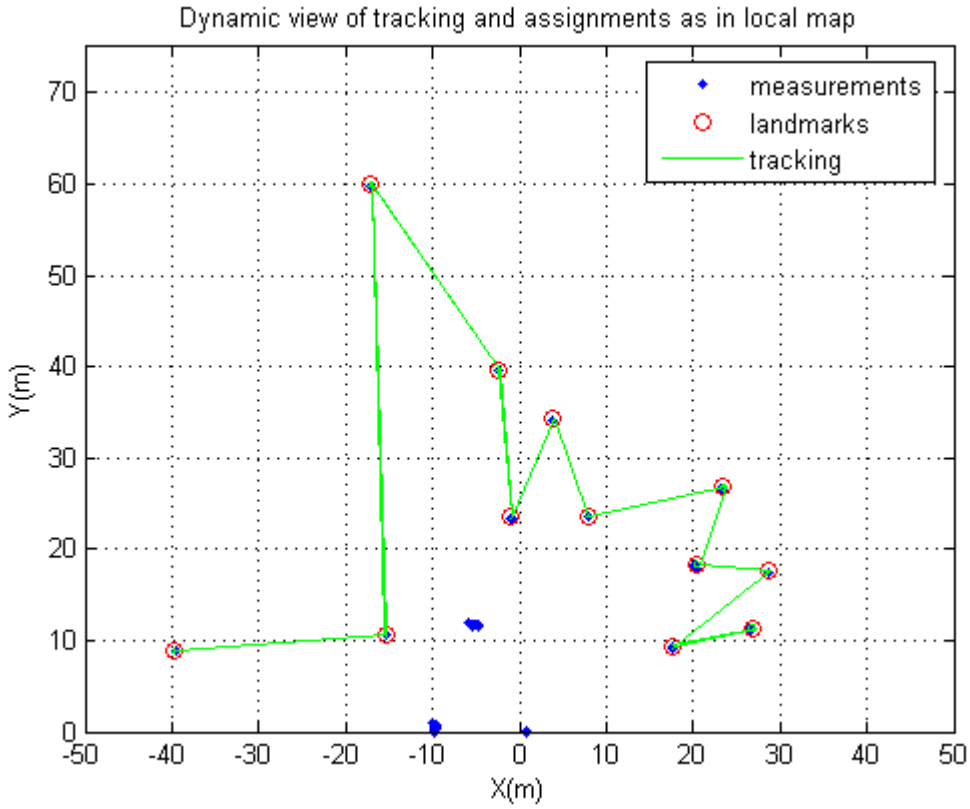


Figure 9-27 Tracking Data Association scenario at scan k= 20

9.4.2 Outdoor scenario performance evaluation

We also test our SLAM data association algorithm in outdoor environment which is the campus of Vaughn College near La Guardia airport. We placed a Garmin ™ GPS receiver on a shopping cart and walked around the periphery of the campus. We then uses this GPS trail data which is later processed through online GPS software called EasyGPS™, as our ground truth to feed into DynaEst for simulation. The GPS data can be mapped onto a Google satellite map which provides the global longitudinal and latitudinal coordinates of each way-points that are either trees or lamp posts alongside of the streets (actually we set it as our landmarks). The Google mapped GPS trajectory is rendered by the online GPS Visualizer™ (by Adam Schneider © 2002-2012) and shown as the figure below:

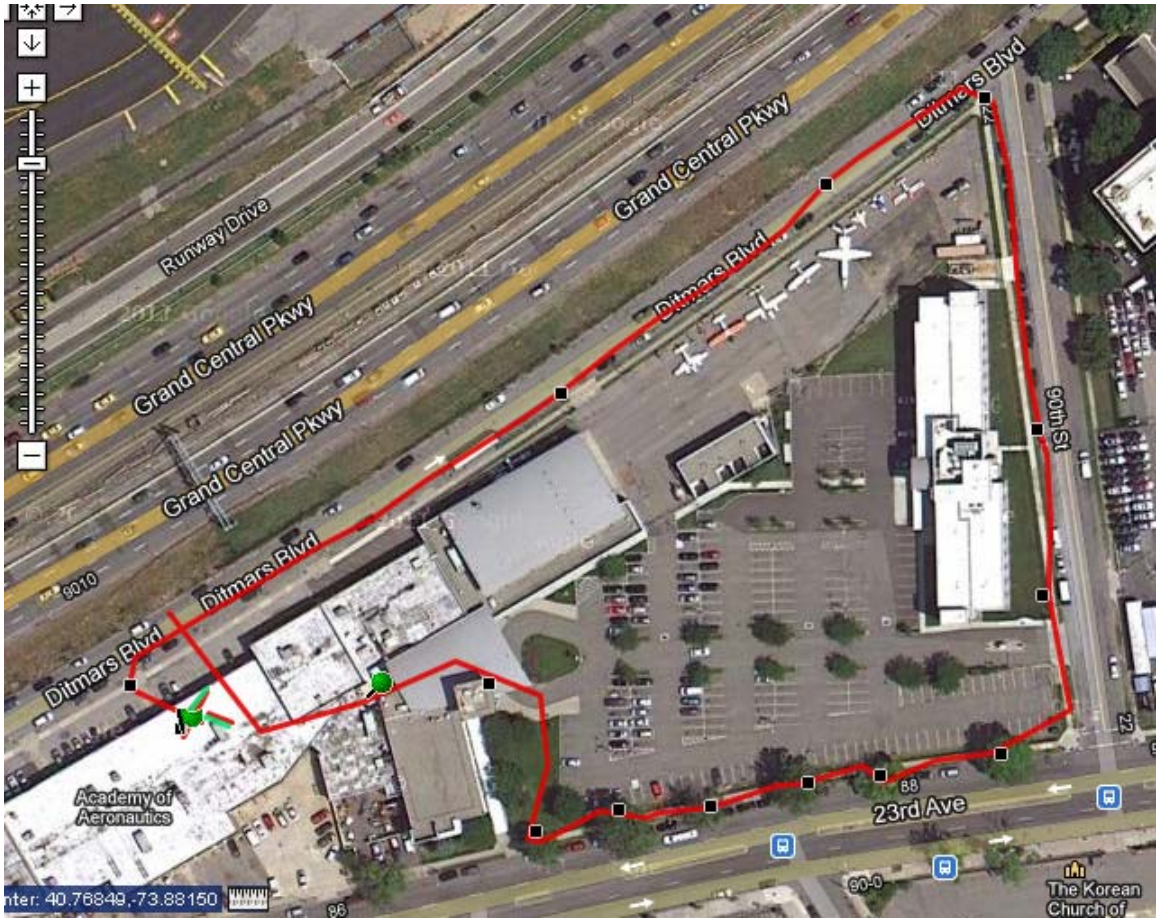


Figure 9-28. The satellite map of GPS path at Vaughn campus.

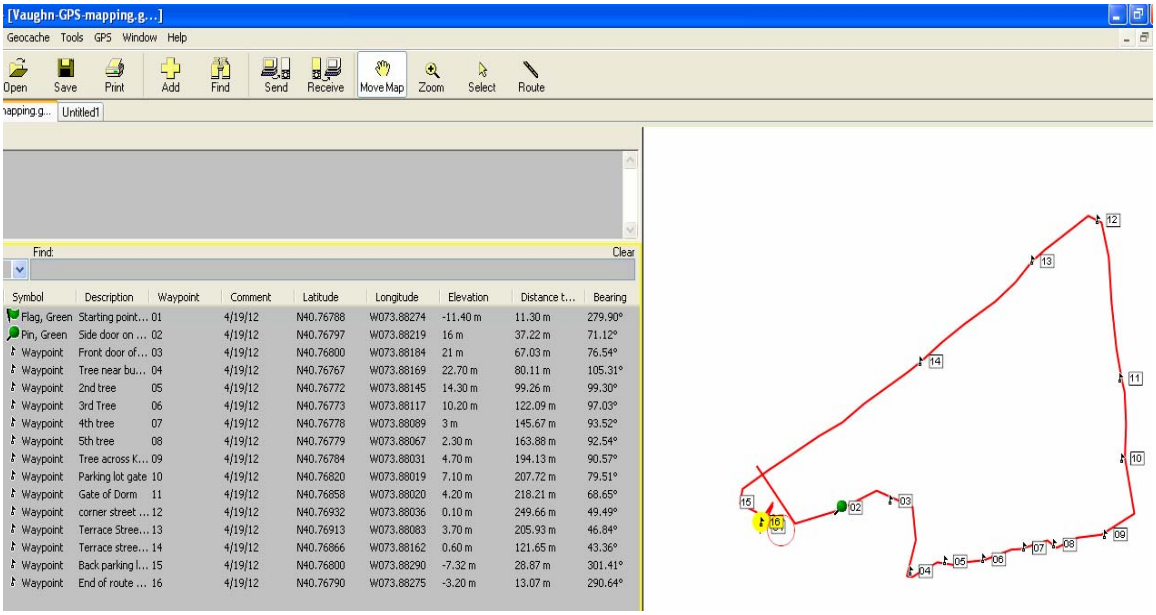


Figure 9-29. The GPS path data processed by EasyGPS to provide the global positions.

We ran the simulations by using the Matlab toolbox of UKF/EKF of SLAM:

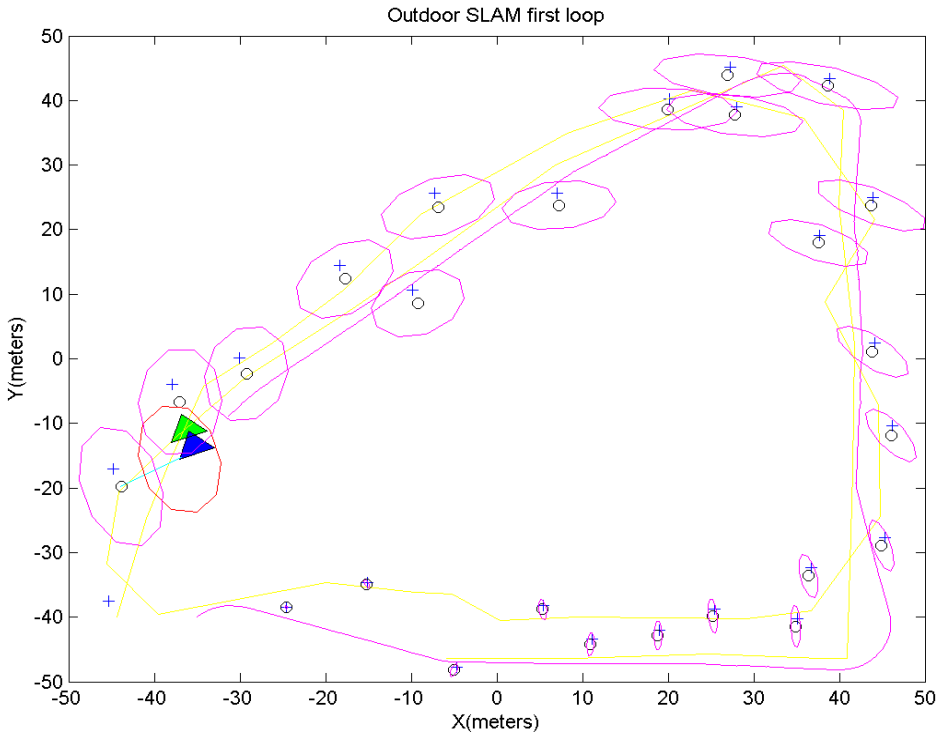


Figure 9-30. The simulated trajectory vs. the GPS ground truth.

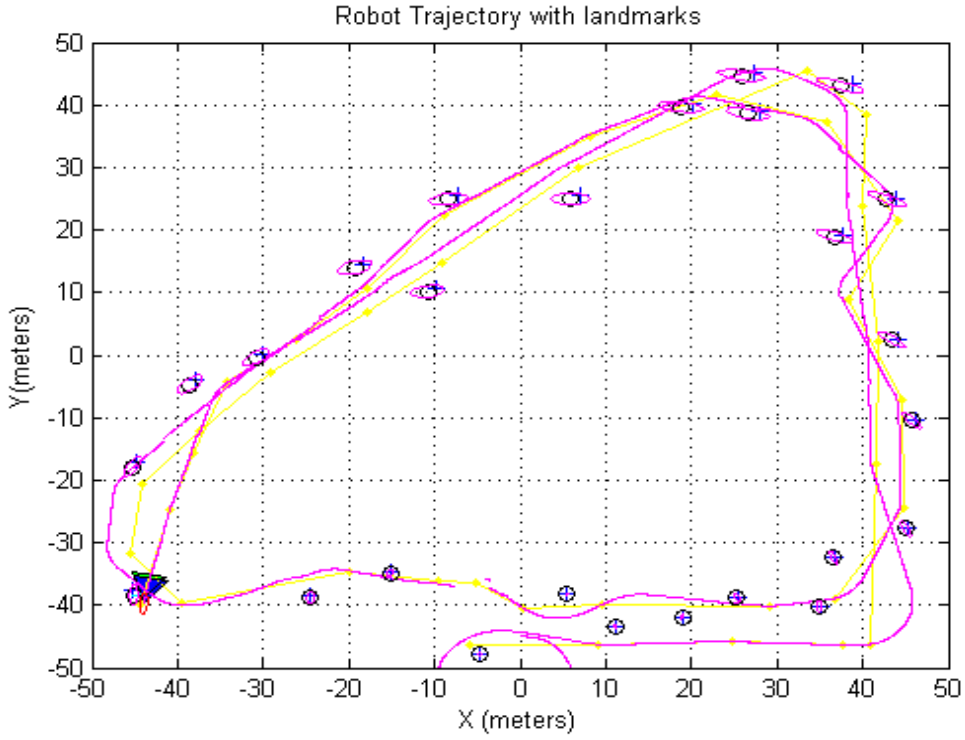


Figure 9-31. The 2nd round of SLAM with reduced landmark state uncertainty.

The result shows the uncertainty of landmark position with larger ellipses in the first loop of trajectory in Figure 9-30, while the uncertainty is much smaller the second round of the traveling loop as shown in Figure 9-31.

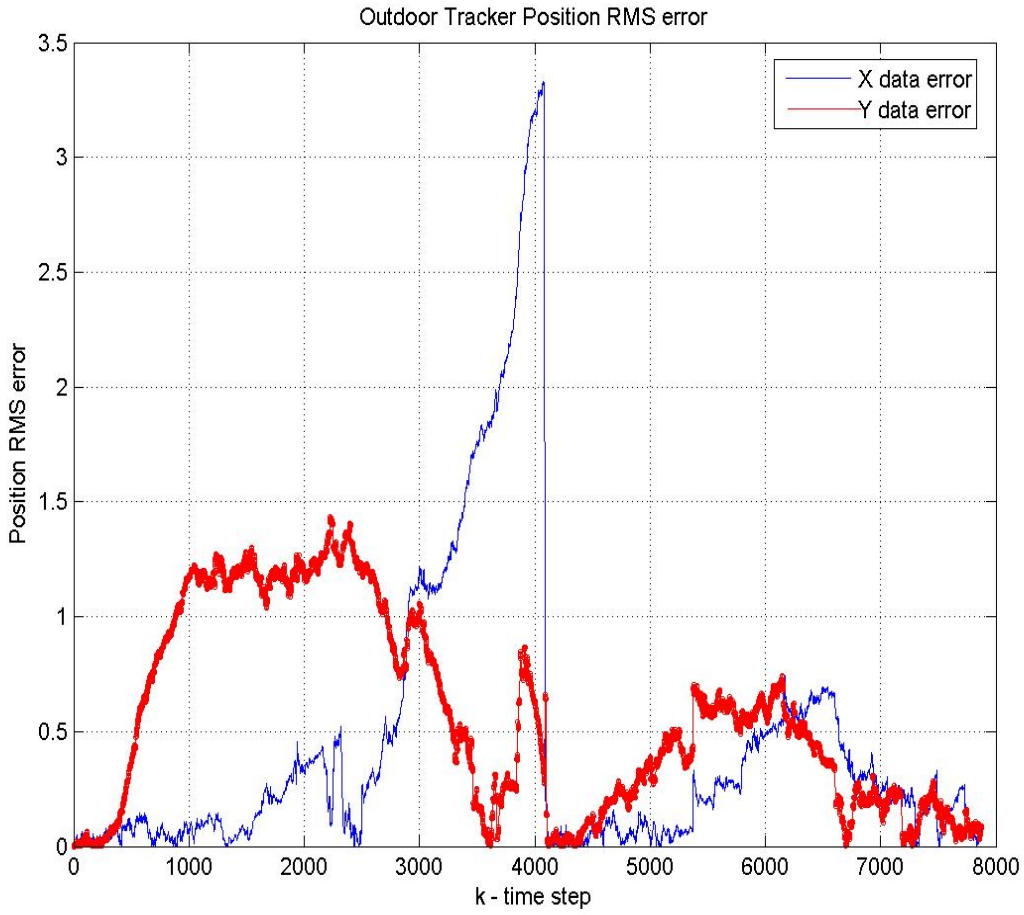


Figure 9-32. The position errors of X, Y coordinates along the trajectory path.

Figure 9-32 displays the position error of x-y coordinates. Notice that the X-coordinate has larger error during the first round of the loop (from k =1 to 4000), and the error is much smaller in the second round of the loop (from k =4001 to 8000). Likewise, for the Y-position error, the error is first round of loop is still larger than the error in the second round of the loop. This shows the data association is more effective if more information is collected when revisiting the landmarks.

Chapter 10

Conclusion and Future work

In this paper, we present the proposed data association for SLAM in a dynamic environment. We first investigate many data association algorithms that have been used in the past and then selectively modify and integrate them into an overall scheme of data association which can be used in the dynamic SLAM system.

10.1 Summary

Based on the concept of interactive multiple models (IMM) and Joint probability data association (JPDA), our algorithm is able to perform data association in spite of the clutter and jamming environment, and quantify its degradation effect for data association. By using M out of N sliding window decision, along with the Q-best 3S-2D assignment optimization, we are able to initialize the new landmark and revert the decision in case of discrepancy occurred in complicate and illusive scenario.

As for the maneuvering objects with acceleration or low varying speed, this scheme is also sufficient to accommodate the variation due to its flexibility in using adaptive model switching mechanism. For some extreme case, such as encountering a running person on the hallway, since robot's low-power sensor may not have the sophisticated signal processing capability, the detected spurious signal may be translated into a sequence of twinkling clutter, and the corresponding measurement would be treated as false alarm by JPDA algorithm. We have verified the performances by running many simulations. Since the DA's task is to associate the measurement with the originating target (landmark or tracked object), once this relationship is confirmed, it will establish the track file on the memory for SLAM to use. We use EKF and UKF as the

tracking filters in order to quickly obtain the metrics such as RMS error, innovation error, filter consistency, and time of convergence to manifest the efficiency of data association algorithms. We did not test this algorithm within the framework of particle filter based SLAM because of the limitation of a small robotic platform. However, we do not mean it cannot be implemented if the processing power is increased and this idea can be adopted in case of distributed computing for networked multiple robots.

Data association is an estimation process which helps in decision making in many fields of applications. It is normally implemented between the prediction process and the update process in Bayesian framework. For SLAM applications, it is up to the SLAM designer to decide whether to use it for update or abandon it, and it entirely depends on the need of applications.

In short, our major contribution in this work is to design an integrated probabilistic filter of data association which works for robotic SLAM operating in an unstructured and dynamic environment where the spurious signals from clutter and interferences are the major problems causing ambiguity in data association. Our filter features in the following capabilities: (1) assuring to provide feasible solutions regardless of adverse situations (2) quickly responding to the dynamic changing environments. (3) ability to revoke the mistaken decision due to the deceptive and confusing phenomena. (4) using only single sensor for data association and SLAM which can be implemented in resource-limited robotic platforms.

10.2 Future work and challenge

Although data association is involved in the update process within a Bayesian estimator, the real effect on SLAM is only marginally addressed. Many assumptions have been

made as default situation. For example, the obstacles which may be used as landmarks are assumed to be solid and opaque. If there is a glass door at the end of the corridor and some posts or corners on the other side of the glass door can be candidates for landmarks because the laser scanner can detect them. But the robot is not supposed to use those landmarks because they are on the other side of the glass door. If the robot uses sonar system for obstacle avoidance, it would turn away from this glass door. So the signal returns from those posts or corners are false alarms. Therefore, even the laser scanner may pick up those signals as echoes from the landmarks; we should be able to distinguish the deceptive returns from the legitimate signal returns of the true landmarks.

The other issue we have not really addressed rigorously is the design of transition probability matrix. We depend on the empirical priors and heuristics to set the probability of model transition. Although this set of parameters has been moderately accepted as the proper values by the tracking community standard, however, the incremental adjustment should be considered and the methodology should be studied further until a feasible solution is obtained. The more acute system response will depend on the development of an algorithm for adaptively chosen probability of model transition.

10.3 The Challenge for dynamic man-machine interaction in the future

Data association has been the key issue in air traffic control community. With the advent of commercial use of the unmanned air vehicles (drones) spawn out in foreseeable days, the air traffic situation becomes more critical and difficult for human operators and the monitoring systems as well. The dense traffic creates more problems in air space traffic and airport management, air to ground communication, aircraft identification and location,

air space security, and many related issues. For land-based autonomous vehicles or robots, this is also an issue due to the intermingling of human and machine working condition. The basic intelligence for smoother cooperation between human and the smart machine is the ability to discern the situation and metaphor for cognition. Data association is the beginning of the artificial intelligence in this regard. Pattern recognition and classification are the next steps of the advanced intelligence which set forth for the machine learning.

In view of the importance of this trend, we will continue our research towards the area of data association in machine learning and operational optimization.

Appendix A: Unscented Kalman Filter (UKF)

For an n-dimensional Gaussian PDF with mean $\bar{x}$ and covariance and Σ_x, there are 2n +1 sigma points, χ^i are chosen based on the following rule:

$$\chi^0 = \bar{x} \ ; \tag{A-1}$$

$$\chi^i = \bar{x} + (\sqrt{(n+\lambda)\Sigma_x} \ ), \text{ for } i = 1, \ldots, \ n; \tag{A-2}$$

$$\chi^i = \bar{x} - (\sqrt{(n+\lambda)\Sigma_x} \ ), \text{ for } i = n + 1, \ldots, \ 2n; \tag{A-3}$$

where $\lambda = \alpha^2 (n + \kappa) - n$, α and κ are scale factors where κ will take effect on higher order moments (i.e., on 4th order). Since the mean is only a function of first two orders, none of properties of higher order moments affect the result. So, we set $\kappa = 0$. Each sigma-point has two weights, one for computing the mean; while the other for recovering the covariance. They are described as follows:

$$W_m^0 = \frac{\lambda}{n+\lambda}; \ W_c^0 = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta); \tag{A-4}$$

$$W_m^i = W_c^i = \frac{1}{2(n+\lambda)}; \text{ for } i = 1 \ldots \ 2n \tag{A-5}$$

where $\alpha = 10^{-3}$, $\beta = 2$ (for example). For the prediction stage, these sigma points are propagated through the transition function

$$y^i = g(\chi^i); \text{ for } i = 1 \ldots \ 2n \tag{A-6}$$

Then the weighted sigma-points are recombined to produce new mean and covariance

$$\hat{x}_{k|k-1} = \sum_{i=0}^{2n} W_m^i y^i \tag{A-7}$$

$$\Sigma_{k|k-1} = \sum W_c^i (y^i - \hat{x}_{k|k-1})(y^i - \hat{x}_{k|k-1})^T + \Sigma_{x-1} \tag{A-8}$$

For the update stage, the predicted mean and covariance are augmented with newly observed features, except now with the mean and covariance of the measurement noise.

$$\hat{x}_{aug} = \left[(\hat{x}_{k|k-1})^T \quad E\left|\delta_k^T\right|\right]^T \; ; \; \Sigma_{aug} = \begin{bmatrix} \Sigma_{k\}k-1} & 0 \\ 0 & Q_k \end{bmatrix} \tag{A-9}$$

Again, a set of 2 n +1 sigma-points is derived from the augmented state where n is the dimension of augmented state. By repeating the previous procedure, except using the newly acquired mean and covariance instead, we have

$$\psi^0 \; = \; \hat{x}_{aug} \tag{A-10}$$

$$\psi^i \; = \; \hat{x}_{aug} \; + (\sqrt{(n+\lambda)\Sigma_{aug}} \;), \text{ for } i = 1, \ldots, \; n \tag{A-11}$$

$$\psi^i \; = \; \hat{x}_{aug} \; - (\sqrt{(n+\lambda)\Sigma_{aug}} \;), \text{ for } i = n + 1, \ldots, \; 2n \tag{A-12}$$

The sigma-points are then projected through the observation function *h*, that is

$$\gamma^i = h(\psi^i) \text{; for } i = 1 \ldots \; 2n. \tag{A-13}$$

The weighted sigma points are recombined to produce the predicted measurement and predicted measurement covariance as follows:

$$\hat{z}_k = \sum_{i=0}^{2n} W_m^i \gamma^i \; ; \; S_k = \sum W_c^i (\gamma^i - \hat{z}_k)(\gamma^i - \hat{z}_k)^T + Q_x \, . \tag{A-14}$$

The cross covariance is obtained as follows:

$$\Sigma_{x,z} = \sum_{i=0}^{2n} W_c^i (\psi^i - \hat{x}_{k|k-1})(\gamma^i - \hat{z}_k)^T \tag{A-15}$$

To compute the Kalman gain, *K*

$$K = \Sigma_{x,z} \, S_k^{-1} \tag{A-16}$$

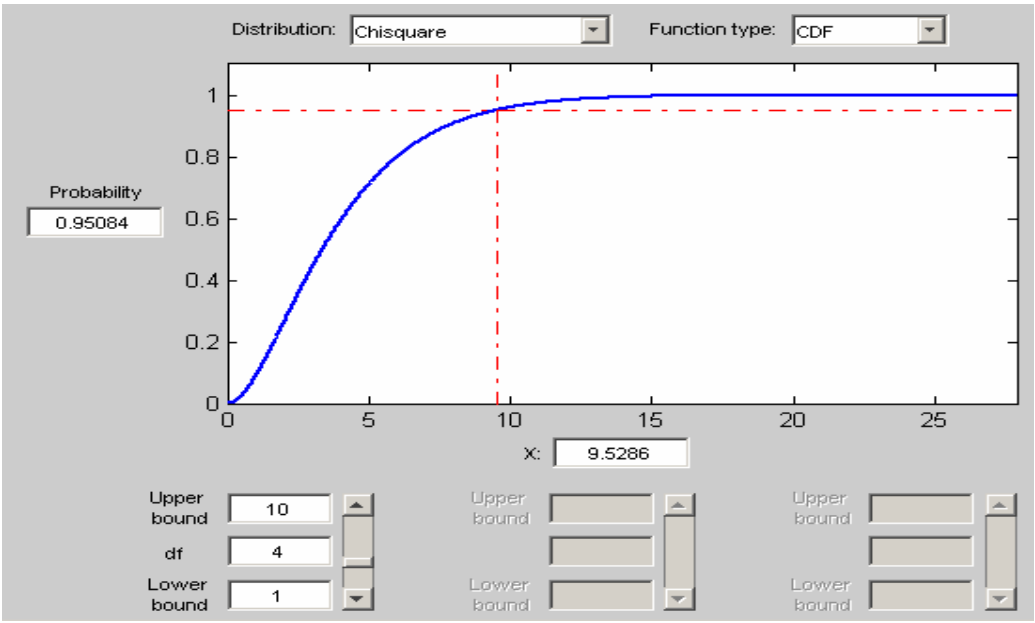
Finally, the update mean and covariance can be obtained as

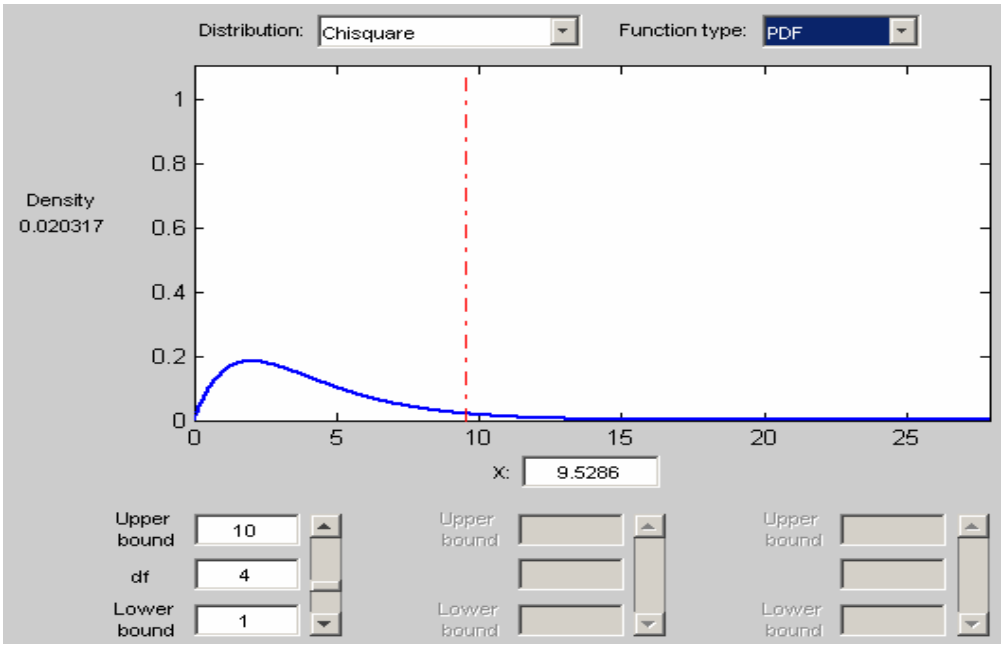
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K(z - \hat{z}) \tag{A-17}$$

$$\Sigma_{k|k} = \Sigma_{k|k-1} - KS_k K^T \tag{A-18}$$

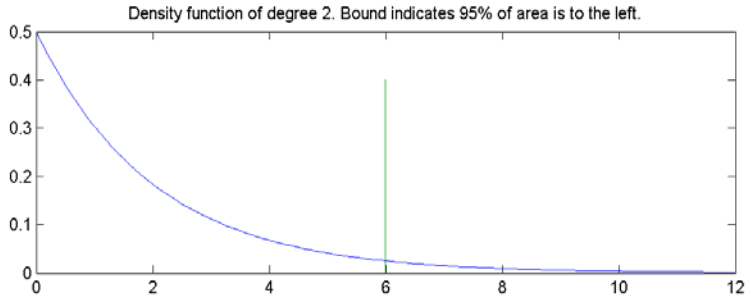
Appendix B: Chi-Square probability distribution function and density function

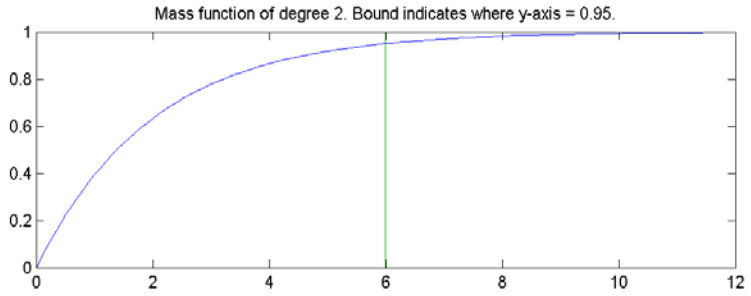
The following Matlab Distribution tool shows Chi-Square CDF/pdf with degree of freedom of 4 and 95% detection probability.

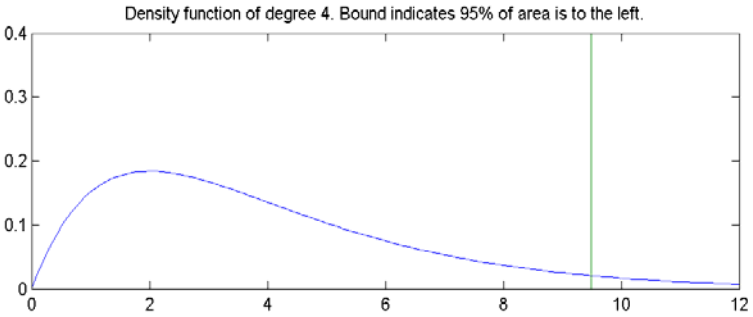


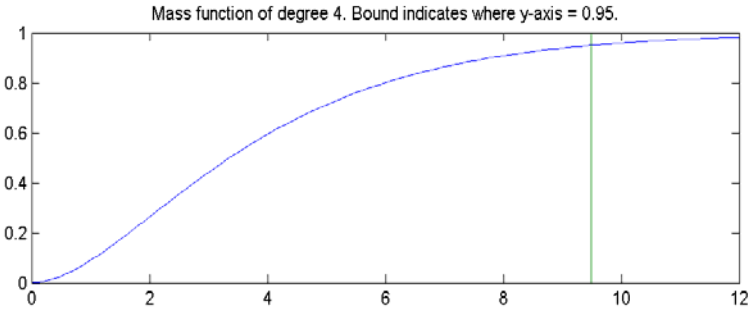


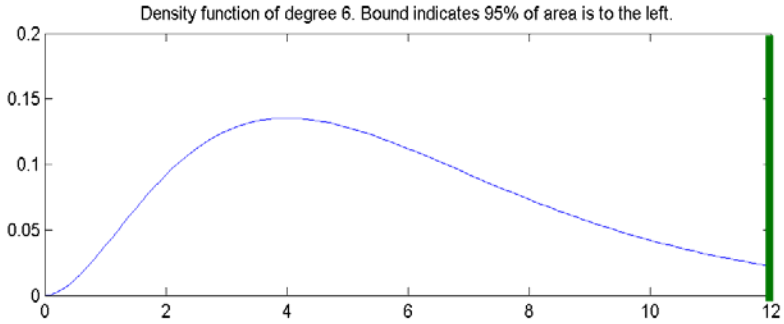
The following figures are listed for different degree of freedom:

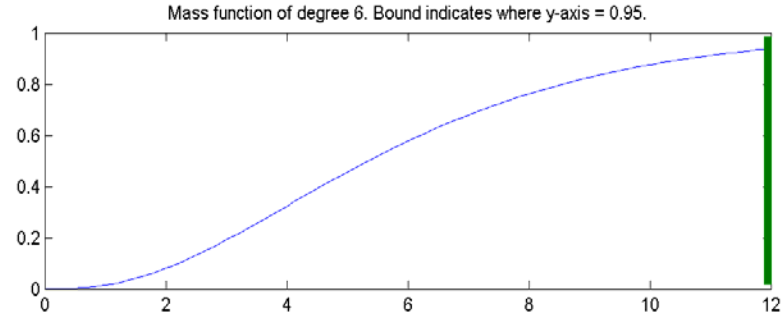












Appendix C: Derivation of q factor.

The scalar degradation factor q_k (its value lies between 0 and 1) is given as

$$q_k \triangleq P_D \frac{c_D}{(2\pi)^{D/2}} \sum_{m=1}^{\infty} \frac{e^{\lambda A}(\lambda A)^{m-1}}{(m-1)!} \left(\frac{D}{gD}\right)^{m-1} I_k(m) \tag{C-1}$$

$$\text{where } I_k(m) \triangleq \int_0^g \cdots \int_0^g \frac{\exp(-r_1^2) r_1^2}{b_k + \sum_{j=1}^{m} \exp(-r_j^2/2)} \times (r_1 r_2 \cdots r_m)^{D-1} dr_1 \cdots dr_m \tag{C-2}$$

where D is dimension of innovation vector; c_D is unit region of D-dimensional region, given as

$$c_D = \pi^{D/2} / \Gamma(\tfrac{D}{2} + 1) \tag{C-3}$$

where $c_1 = 2$; $c_2 = \pi$; $c_3 = 4\pi/3$. …etc. g is g-sigma gate, which equals to square root of validation threshold, $\gamma^{1/2}$. Γ is the gamma function. The symbol b_k is given as

$$b_k = (2\pi)^{D/2} (\lambda A / c_D g^D)(1 - P_D)/P_D \tag{C-4}$$

From above derivation, we can see the degradation factor, q_k, is explicitly dependent on the probability of detection and the occurrence of false alarms within the validation gates. q_k reduces the covariance improvement; the smaller q_k is, the more degradation (in the worst case, $q_k = 0$).

Appendix D: Partial Matlab code for SLAM Data Association simulation

```
% (1) SLAM data association code for simulation.
function data = Thesis_Indoor(lm, wp)
format compact
configfile; % ** USE THIS FILE TO CONFIGURE SLAM **
************************************************
% Setup plots
fig = figure;
plot(lm(1,:),lm(2,:),'b+')
hold on, axis([-50,50,-50,50])
% plot(wp(1,:),wp(2,:), 'g', wp(1,:),wp(2,:),'g.')
plot(wp(1,:),wp(2,:), 'y', wp(1,:),wp(2,:),'y-')
************************************************
% Initialise states and other global variables
global RX GX PX DATA
GX = [0 0 pi/4]'; % Ground Truth
RX = [0 0 pi/4]'; % Robot path
PX= eye(3)*eps; % Covariance Sigma
DATA= initialise_store(RX,GX,PX); % stored data for off-line
************************************************
xlabel ('meters'), ylabel ('meters')
set (fig, 'name', 'UKF-SLAM Data Association')
% title ('GNN DA in dense but clutter-free environment')
h = setup_animations;
veh = [0 -WHEELBASE -WHEELBASE; 0 -2 2]; % for vehicle animation
plines = [ ]; % for laser line animation
************************************************
% Main loop
tic  % begin timing
while iwp ~= 0
   % Compute true data
[G,iwp]= compute_steering (xtrue, wp, iwp, AT_WAYPOINT, G, RATEG, MAXG,  dt);
   if iwp == 0 & NUMBER_LOOPS > 1, pack; iwp=1; NUMBER_LOOPS=
NUMBER_LOOPS-1; end % perform loops: if final waypoint reached, go back to
firstxtrue= vehicle_model(xtrue, V,G, WHEELBASE,dt);
[Vn,Gn]= add_control_noise(V,G,Q, SWITCH_CONTROL_NOISE);
************************************************
% UKF predict step
   predict (Vn,Gn,QE, WHEELBASE,dt);
% Incorporate observation, (available every DT_OBSERVE seconds)
   dtsum= dtsum + dt;
   if dtsum >= DT_OBSERVE
      dtsum= 0;
 % Compute true data
      [z,ftag_visible] = get_observations (xtrue, lm, ftag, MAX_RANGE);
      z = add_observation_noise (z,R, SWITCH_SENSOR_NOISE);
```

```
      % UKF update step
  [zf,idf, zn, da_table]=  data_associate_known(XX,z,ftag_visible, da_table);
  if data association is unknown, then
      [zf,idf,zn, da_table] = data_associate_unknown(XX,z,ftag_visible, da_table);
      update(zf,RE,idf);
      augment(zn,RE);
  end
*************************************************************
function h = setup_animations()
*************************************************************
function p = make_laser_lines (rb,xv)
*************************************************************
function p = make_vehicle_covariance_ellipse(x,P)
*************************************************************
function p = make_feature_covariance_ellipses(x,P)
% compute ellipses for plotting feature covariances
*************************************************************
function data = finalise_data(data)
% offline storage finalisation
data.path = data.path(:,1:data.i);
data.true = data.true(:,1:data.i);
```

(2) %Rex's IMM simulation

```
% Space for model parameters
index = cell(1,nmodels); F   = cell(1,nmodels); L   = cell(1,nmodels); Qc  =
cell(1,nmodels);
A   = cell(1,nmodels); Q   = cell(1,nmodels); H   = cell(1,nmodels); R   = cell(1,nmodels);
% Index vector of model 1
index {1} = [1 2 3 4]';
% Transition matrix for the continous-time velocity model.
F{1} = [0 0 1 0; 0 0 0 1; 0 0 0 0; 0 0 0 0];
% Noise effect matrix for the continous-time system.
L{1} = [0 0; 0 0; 1 0; 0 1];
% Process noise variance
q1 = 0.01; Qc{1} = diag ([q1 q1]);
% Discretization of the continous-time system.
[A{1},Q{1}] = lti_disc(F{1},L{1},Qc{1},dt);
% Process noise variance
KF_q1 = .05; KF_Qc1 = diag([KF_q1 KF_q1]);
% Discretization of the continous-time system.
[KF_A1,KF_Q1] = lti_disc(F{1},L{1},KF_Qc1,dt);
%Specification of the 2nd & 3rd models (turning models)
% The difference is the sign of turning rate....(+ left, - right)
% Measurement models.
H{1} = [1 0 0 0;  0 1 0 0]; H{2} = [1 0 0 0 0; 0 1 0 0 0]; hdims = 2;
```

```
% probability of model Transition
p_ij  = [0.90 0.05  0.05; 0.05 0.90 0.05; 0.05 0.05 0.9]; % Transition probability.
% Generate object state values
for i = 2:n
  st = mstate(i);
  if isstr(a_func{st}) | strcmp(class(a_func{st}),'function_handle')
     X_r(ind{st},i) = feval(a_func{st},X_r(ind{st},i-1),a_param{st});
  else
     X_r(ind{st},i) = A{st}*X_r(ind{st},i-1);
  end
end
% Noise variances of the measurement models

% Model 1
r1 = .05; r2 = .05; R{1} = diag([r1 r2]);
% Model 2 & 3
r1 = .1; r2 = .1; R{2} = diag([r1 r2]);
% Generate the simulated landmarks.
LM = zeros(hdims,n);
for i = 1:n
%     j = i*3;
%     if i == j then
       LM(:,i) = H{mstate(i)}*X_r(ind{mstate(i)},i)+ gauss_rnd(zeros(size(LM,1),1),
R{mstate(i)});
%     else
%       LM(:,i)=[0 0]';
%     end
end
% Calculate the MSEs
MSE_KF1_1 = mean((X_r(1,:)-KF_MM(1,:)).^2);
MSE_KF1_2 = mean((X_r(2,:)-KF_MM(2,:)).^2);
MSE_KF1 = 1/2*(MSE_KF1_1 + MSE_KF1_2);
MSE_EIMM1 = mean((X_r(1,:)-EIMM_MM(1,:)).^2);
MSE_EIMM2 = mean((X_r(2,:)-EIMM_MM(2,:)).^2);
MSE_EIMM = 1/2*(MSE_EIMM1 + MSE_EIMM2);
MSE_UIMM1 = mean((X_r(1,:)-UIMM_MM(1,:)).^2);
MSE_UIMM2 = mean((X_r(2,:)-UIMM_MM(2,:)).^2);
MSE_UIMM = 1/2*(MSE_UIMM1 + MSE_UIMM2);
```

List of Publications

The following research papers are submitted to the technical conferences and published in their related proceedings and journals during my study as doctoral student:

1. R. Wong, J. Xiao, and S. Joseph, "An adaptive data association for robotic SLAM in search and rescue operation." in Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, China, August 2011. pp 997-1003. Digital Object Identifier: 10.1109/ICMA.2011.5985796
2. R. Wong, J. Xiao, and S. Joseph, "Clutter-sensitive data association for simultaneous localization and mapping in robotic wireless sensor networks" in Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA), Xian, China, August 2010. pp 1976-1981. DOI: 10.1109/ICMA.2010.5587987
3. R. Wong, J. Xiao, S. Joseph, and Z. Shan, "Data Association for Simultaneous Localization and Mapping in robotic wireless sensor networks." in Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Montreal, Canada, July 2010, pp 459-464, DOI: 10.1109/AIM.2010.5695906
4. R. Wong, J. Xiao, and S. Joseph, "A Robust Data Association for Simultaneous Localization and Mapping in Dynamic Environments." in Proceedings of the IEEE International Conference on Information and Automation (ICIA), Harbin, China, June 2010, pp 470~475. DOI:10.1109/ICINFA.2010.5512382 (Finalist Award of Best Paper)
5. R. Wong, S. Hu, F. Cabrera-Mora, J. Xiao; J. Tan, "A Distributed algorithm for mobile robot localization and mapping in wireless sensor networks." in Proceedings

of the IEEE International Conference on Information and Automation (ICIA), Changsha, China, June 2008, pp 560-566. DOI: 10.1109/ICINFA.2008.4608063

6. R. Wong, J. Xiao, and S. Joseph, "A Robust Data Association for Simultaneous Localization and Mapping in Dynamic Environments." published in the Special Issues on Information, an International Interdisciplinary Journal in English, Japanese and Chinese by International Information Institute (III), Vol.13, No.6, November, 2010. ISSN 1343-4500(print), ISSN 1344-8994(electronic)
7. R. Wong, J. Xiao, and S. Joseph, "A Mixed Model Data Association for Simultaneous Localization and Mapping in Dynamic Environments" submitted to International Journal of Mechatronics and Automation on April 1, 2012. Submission Code: IJMA36446. Pending for review.

Bibliography

Akdere, M. "Continuous Probabilistic Data Association with Constraints", CS Journal at Brown,2009

Armesto, L., Ippoliti, G., Longhi, S., and Tornero, J. "Probabilistic Self-Localization and Mapping", IEEE Robotics & Automation, Aug. 2008

Arulampalam, S. , S. Maskell, N. Gordon and T. Clapp, “Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayesian Tracking”,2001

Bailey, T., “ Mobile Robot Localization and Mapping in Extensive Outdoor Environments”, Doctoral thesis, University of Sydney, 2002

Bertsekas, D. “The Auction Algorithm for Assignment and Other Network Flow Problems: A Tutorial”, Interfaces, Vol. 20, pp. 133-149. 1990

Bertsekas, D., "The auction algorithm: a distributed relaxation method for the Assignment problem", Annals of Operations Research, Volume 14, Number 1 (1988), 105-123, 1987

Bibby, J, I. Reid, “Simultaneous Localization and Mapping in Dynamic Environment with Reverse Data Association”, IEEE Proc. Robotics: Science and Systems Conference, Atlanta, June 2007

Blom, H., I. Reid, “Simultaneousl Localization and Mapping in Dynamic Environment with Reverse Data Association”, IEEE 2008

Blom, H., and Shalom, Y.B., “The interacting multiple model algorithm for systems with Markovian switching coefficients”. IEEE Transactions on Automatic Control, 33(8):780–783. 1988

Bosse, M.C., “ATLAS:A framework for large scale automated Mapping and Localization’, MIT Press, 2004

Bourgeois, F., “An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices”, Communications of ACM. 1971

Castellanos, J., M. Devy, J. Tardos, “Towards a topological representation of indoor environment: a landmark based approach”, IEEE/RSJ IROS, 1999.

Civera, J., Davidson, A., and Montiel, J. "Interactive Multiple Model Monocular SLAM", IEEE/ICRA, 2008

Cox, I., and Leonard, J." Probabilistic Data Association for Dynamic World Modeling:

A Multiple Hypothesis Approach. IEEE/AES, 1991

Cox, I,, M. Miller, " On finding ranked assignment with application to multi-target tracking and motion correspondence", IEEE Transaction on Aerospace and Electronic Systems, 1995

Crisan, D., and Doucet, A "A survey of convergence results on particle filtering methods for practitioners", IEEE Transactions on Signal Processing 2002

Csorba, M., M. "Simultaneous Localization and map Building." PhD thesis, University of Oxford, Dept. of Engineering Science. 1997.

Davey, S. "SLAM building Using the Probabilistic Multi-Hypothesis Tracker", IEEE Trans on Robotics, Vol 23, No.2, April 2007

Daum, F, "Book Review on Multiple-Multisensor Tracking: Principles and Techniques", IEEE/AES Systems Magazine, 1996

Dijkstra, E.W.," A note on two problems in connection with graph.", Numerische Mathematik, pp 269-271, 1959

Dissanayake, M., P. Newman, S. Clark, H.F. Durrant-Whyte, and Csorba. "Solution to the Simultaneous Localization and Map Building (SLAM) problem". IEEETransactions on Robotics and Automation, 17(3):229–241, 2001.

Doucet, A N de Freitas and N Gordon. "Sequential Monte Carlo Methods in Practice", Springer, 2001

Feder, H., J. Leonard, "Adaptive mobile robot navigation and mapping". International Journal of Robotic Research, 1999.

Fitzgerald, R. "Track Bias and Coalescence with Probabilistic Data Association", IEEE Trans. AES, Vol.21, No.6, pp822-830. 1985

Fitzgerald, R. "Development of practical PDA logic for multitarget tracking by microprocessor: Multitarget-Multisensor Tracking: Advanced Applications", Chapter 1, Artech House. 1990

Fortmann, T,, Shalom, Y.B. and Scheffe, M. "Sonar tracking of multiple targets using Joint Probabilistic Data Association", IEEE Journal of Ocean Engineering, Vol. OE-9, No.3., 1983

Gordon, N.J., D. Salmond, and A.Smith,"Novel approach to non-linear/non-Gaussian Bayesian state estimation". IEEE proceeding-F, 1993

Grisettia, G, Tipaldib, G.D., Stachnissc, C., Burgarda, W., Nardib, D., "Fast and accurate

SLAM with Rao–Blackwellized particle filters", Robotics and Autonomous Systems 55 pp30–38, 2007

Gutmann, J.S. and K. Konolige. Incremental mapping of large cyclic environments. In Proc. IEEE International Symposium on Computational Intelligence in Robotics and Automation, pages 318–325. IEEE, 2000.

Hartikainen, J., Matlab toolbox EKF/UKF, 2011, http://www.lce.hut.fi/research/mm/ekfukf/

Jonker, R.,. A shortest Augmenting Path Algorithm for Dense and Sparse Linear Assignment Problems", Journal of Computing, Vol.38, pp-325-340. 1987

Julier, S., J. Simon J. and Jeffery K. Uhlmann.1997. A New Extension of the Kalman Filter to nonlinear Systems. In The Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II, SPIE.

Julier, S. J., J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation", Proceedings of The IEEE, Vol, 92, No.3, March 2004

Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems, Transactions of the ASME - Journal of Basic Engineering Vol. 82: pp. 35-45. 1960.

Kalman, R. E., Bucy R. S. 1961. "New Results in Linear Filtering and Prediction Theory", Transactions of the ASME - Journal of Basic Engineering Vol. 83: pp. 95-107.

Knight, J., A. Davison, and I. Reid, "Towards Constant Time SLAM using Postponement", IEEE/IROS, 2001

Kragel, B., Hermman, S., and Roseveare, N. "A comparison of Methods for Estimating Track to Track Assignment Probabilities", IEEE/AES 2010

Leung, H." Evaluation of Multiple Radar Target Trackers in Stressful Environments", IEEE Trans. /AES. 1999

Li, R.X. "Survey of Maneuvering Target Tracking. Part V: Multiple-Model Methods", IEEE Trans/AES. 2005

Miller, M. "Optimizing Murty's Ranked Assignment Method", IEEE Trans/AES, Vol. 33. April 2001

Montemerlo, M., S. Thrun, "Fast SLAM", Springer, 2007

Murty, K. “An algorithm for ranking all the assignments in order of increasing cost”, JSTOR: Operation Research, Vol. 16 No.3, pp. 682-687. 1968

Nagarajan, V. “Combinatorial Problem in Multitarget Tracking”, IEEE Proceeding, 1987

Newman, P. “On The Structure and Solution of the Simultaneous Localization and Map Building Problem”. PhD thesis, University of Sydney, Australian Centre for Field Robotics. 1999

Niera, J. and Tardos, J. ”Data Association in Stochastic Mapping using the Joint Compatibility Test”, IEEE/IROS. 2002

Niera, J. and J.Tardos, “Data Association in Stochastic Mapping using the Joint Compatility Test”, IEEE, ICRA, 2000

Nieto, J., Guivant, J., and Nebot, E., "Real Time Data Association for FastSLAM", ICRA, 2003

Popp, R. “m-Best S-D Assignment Algorithm with Application to Multitarget Tracking”, IEEE/AES, Jan. 2001

Reid, D. “An Algorithm for Tracking Multiple Targets”, IEEE Trans Automatic Control. 1979

Ristic, B, S. Arulampalam, N. Gordon, “Beyond the Kalman Filter: Particle Filters for Tracking Applications”, Artech House, 2004.

Rodriguez-Losada, D. F. Matia, A. Jimenez, “Local maps fusion for real time multirobot indoor simultaneous localization and mapping”, in Proceedings of the 2004 IEEE International Conference on Robotics and Automation, 2004

Roecker, J. “Suboptimal Joint Probabilistic Data Association”, IEEE Trans. AES, Vol. 29, No.2, pp510-517. 1993

Shalom, Y. B. and Fortmann, T. “Tracking and data association”, Academic Press, pp.9-20. 1988

Shalom, Y.B., Li, R.X. and Kirubarajan, “Estimation with Applications to tracking and Navigation”, John Wiley, pp.441-470. 2001

Shalom, Y.B. and T. Fortmman, “Tracking and Data association” , Academic Press, Inc., 1988.

Shalom, Y.B. and Li, R.X. “Multitarget-Multisensor tracking: principle and techniques”, YBS publishing, pp.499-510. 1995

Shalom, Y.B. “DynaEstTM 3.35”, 04/03/2001, Copyright (c) 2000. A companion CD with the text "Estimation with Applications to tracking and Navigation", John Wiley, 2001.

Siegwart, R. and I.R. Nourbakhsh, “Intorduction to Autonomous Mobile Robots”, MIT press, 2004.

Simhon, S., G. Dudek, “A global topological map formed by local metric maps”, IEEE/RSJ, IROS, p 1708-1716, 1998

Sittler R.W. “An Optimal Data Association Problem in the Surveillance Theory”, IEEE Trans. Military Electronics. 1964

Smith, R., Self, M, and Cheeseman, P. “Estimating uncertain spatial relationships in robotics. Autonomous Robot Vehicles”, pages 167–193, 1990

Smith, R., Self, M., and Cheeseman, P.”Stochastic map for uncertain spatial relationships, Autonomous Mobile Robots: Perception, Mapping and Navigation, 1:323–330, 1987

Stone, L., C. Barlow, and T. Corwin, "Bayersian Multiple Target Tracking” Artech House, 1999.

Tardos,T., J. Niera, P. Newman, J. Leonard, “Robust mapping and localization in indoor environment using sonar data”, International Journal of Robotics Research, 2002.

Taylor, J., "The Cramer-Rao estimation error lower bound computation for deterministic nonlinear systems,", IEEE Trans. Automatic Control, Vol. 24, pp.343-344, April 1979.

Thrun, S., Fox, D., and Burgard, W. “A probabilistic approach to concurrent mapping and localization for mobile robots”. Machine Learning and Autonomous Robots (joint issue). 1998

Thrun, S., Fox, D., Burgard, W., and Dellaert, F. ’Robust Monte Carlo Localization for mobile robots". Artificial Intelligence. 2000

Thrun, S., Fox, D., and Burgard, W. “Probabilistic Robotics”, MIT press. 2005

Thrun,S. “Robotic mapping: A survey”, in “Exploring Artificial Intelligence in the New Millenium”, G. Lakemeyer and B. Nebel, editors,. Morgan Kaufmann, 2002.

Uhlmann, J. “Dynamic map building and localization”, PhD thesis, University Oxford,

Dept. of Engineering Science, 1995

Wijesoma, W. “Toward Multidimensional Assignment Data Association in Robot Localization and Mapping”, IEEE Transaction on Robotics. 2004

Williams, S., “Efficient Solutions to Autonomous Mapping and Navigation Problems”, PhD Dissertation, University of Sydney. 2001

Zhou, B. and Bose, N. “Multitarget Tracking in clutter: Fast Algorithm for Data Association”, IEEE Trans. on Aerospace and Electronics. 1993