CHAPTER 5

SELF-LOCALIZATION AND POSE ESTIMATION
OF MOBILE ROBOT

5.1 SELF LOCALIZATION

An accurate determination of location is the fundamental requirement for mobile robots to move in their workspaces. The focus of this work is on the localization problem of determining the position of a mobile robot in an unknown environment as accurately as possible. Most existing localization approaches are passive, i.e. they do not exploit the opportunity to control the robot’s effectors during localization. We propose an active localization approach which is based on probabilistic localization (Leopoldo Armesto et al 2008) and provides rational criteria for setting the robot’s motion direction (exploration), and determining the pointing direction of the sensors so as to most efficiently localize the robot. Furthermore, it will also be able to deal with noisy sensors and approximate world models. In this system a multi sensor integration (Castellanos et al 2001), using simple and inexpensive map based sensor processing, which allows a mobile robot to understand the environment and convey its belief about the environment to a remote location. The work was done in two phases; first phase is to formulate an optimal path planning algorithm (Nakju Lett Doh et al 2007) and usage of a mobile robot simulator which can simulate the above said aspects by coding. The second phase includes implementation of the results using a mobile robot in an unstructured office environment.
5.1.1 Background

Mobile robot navigation is the mechanisms that allow a mobile robot to move through a real world environment to perform its tasks—including locomotion, sensing, localization, and motion planning. The design of any successful robot involves the integration of many different disciplines, among them kinematics, signal analysis, information theory, artificial intelligence, and probability theory (Borenstein and Feng 1996). The problem of navigating and localizing a robot in both known and unknown environment has been studied in many researchers. For autonomous mobile robots, the ability to function and interact with an unknown environment is of key importance. For this reason mapping has been an extremely active research area in robotics and artificial intelligence for at least two decades. Robotic mapping addresses the problem of acquiring spatial models of physical environments through mobile robots and it is important for several reasons. Primarily, an accurate model of a robot's surrounding environment enables it to complete many complex tasks more quickly and reliably than without such a model. Another advantage of map building is that it potentially provides a good framework for integrating information from many different sensors, from many different positions and directions, into a single knowledge source, from which complex plans can be created. Using the Simultaneous Localization and Map-Building (SLAM) method (Montemerlo et al 2003), a map is build as the robot navigates an environment, while a localization routine runs in parallel, continually updating and correcting the robot’s estimated pose. SLAM allows robots to operate in an environment without a priori knowledge of a map and without access to independent position information; it makes a robot truly autonomous.
5.1.2 Methodology

One of the main contributions of this work is related to the multirate (Lee and Tomizuka 2003) asynchronous filtering approach for the SLAM (Dissanayake et al 2001) problem. Previous multirate filter contributions are mainly for linear systems. In Tornero (2001) and Colaneri and de Nicolao (1995), a Kalman filter was applied for linear quadratic regulator (LQG) control, while Kalman filter was developed using lifting techniques. The problem of multirate filtering arises from the fact that sensors and actuators of robots are sampled at different sampling rates due to technological limitations, communication channels, processing time, etc (Hyunjeong Lee et al 2007). In this work, significant improvements for robot pose estimation are proposed by introducing multirate techniques to Fast SLAM. Additionally, an optimal pose estimation algorithm is proposed. The methods on Particle Filters such as Fast SLAM (Montemerlo et al 2003) and Rao-Blackwellized particle filters are taken as reference to our work. Moreover, we developed an asynchronous filtering method to deal with measurements of sensors at different sampling rates. One key point is the asynchronous execution of prediction and update steps in the filtering method, which aims to maintain a good system performance. The prediction step is executed within a pre-specified sampling period (generally at a fast sampling rate to reduce discretization errors), and the update step is executed only when measurements are asynchronously received. The tangent bug algorithm is selected for the implementation of the path planning of mobile robot.

5.2 CHALLENGES IN NAVIGATION

Navigation is one of the most challenging competences required of a mobile robot. Success in navigation requires success of the four building blocks of navigation: perception, the robot must interpret its sensors to extract meaningful data; localization, the robot must determine its position in the
environment (Figure 5.1); cognition, the robot must decide how to act to achieve its goals; and motion control, the robot must modulate its motor outputs to achieve the desired trajectory. Of these four components (Figure 5.2), localization has received the greatest research attention in the past decade and, as a result, significant advances have been made on this area.

5.2.1 Sensor Noise

Sensors are the fundamental robot inputs for the process of perception, and therefore the degree to which sensors can discriminate the world state is critical. Sensor noise induces a limitation on the consistency of sensor readings in the same environmental state and, therefore, on the number of useful bits available from each sensor reading. Often, the source of sensor noise problems is that some environmental features are not captured by the robot’s representation and are thus overlooked. For example, a vision system used for indoor navigation in an office building may use the color values detected by its color CCD camera. When the sun is hidden by clouds, the illumination of the building’s interior changes because of the windows
throughout the building. As a result, hue values are not constant. The color CCD appears noisy from the robot’s perspective as if subjected to random error, and the hue values obtained from the CCD camera will not be usable, unless the robot is able to note the position of the sun and clouds in its representation. Illumination dependence is only one example of the apparent noise in a vision-based sensor system. Picture jitter, signal gain, blooming, and blurring are all additional sources of noise, potentially reducing the useful content of a color video image. Consider the noise level (i.e., apparent random error) of ultrasonic range-measuring sensors (e.g., sonars).

When a sonar transducer emits sound toward a relatively smooth and angled surface, much of the signal will coherently reflect away, failing to generate a return echo. Depending on the material characteristics, a small amount of energy may return a small quantity. When this level is close to the gain threshold of the sonar sensor, then the sonar will, at times, succeed or, at other times, fail to detect the object. From the robot’s perspective, a virtually unchanged environmental state will result in two different possible sonar readings: one short and one long. The poor signal-to-noise ratio of a sonar sensor is further confounded by interference between multiple sonar emitters. Often, research robots will have twelve to fourty eight sonars on a single platform. In acoustically reflective environments, multipath interference can occur between the sonar emissions of one transducer and the echo detection circuitry of another transducer. The result can be dramatically large errors (i.e., underestimation) in ranging values due to a set of coincidental angles. Clearly, the solution is to take multiple readings into account, employing temporal fusion or multisensor fusion to increase the overall information content of the robot’s inputs.
5.2.2 Sensor Aliasing

A second shortcoming of mobile robot sensors causes them to yield little information content, further exacerbating the problem of perception and, thus, localization. The problem, known as sensor aliasing, is a phenomenon that humans rarely encounter. The human sensory system, particularly the visual system, tends to receive unique inputs in each unique local state. In other words, every different place looks different. The power of this unique mapping is only apparent when one considers situations where this fails to hold. Consider moving through an unfamiliar building that is completely dark. When the visual system sees only black, one’s localization system quickly degrades. Another useful example is that of a human-sized maze made from tall hedges. Such mazes have been created for centuries, and humans find
them extremely difficult to solve without landmarks or clues because, without visual uniqueness, human localization competence degrades rapidly. In robots, the un的独特性 of sensor readings, or sensor aliasing, is the norm and not the exception. Consider a narrow-beam rangefinder such as an ultrasonic or infrared rangefinder. This sensor provides range information in a single direction without any additional data regarding material composition such as color, texture, and hardness. Even for a robot with many similar sensors in an array, there are a variety of environmental states that would trigger the same sensor values across the array. Formally, there is a many-to-one mapping from environmental states to the robot’s perceptual inputs. Thus, the robot’s percepts cannot distinguish from these states. A classic problem with sonar based robots involves distinguishing between humans and inanimate objects in an indoor setting. When facing an apparent obstacle facing, should the robot say “Excuse me” because the obstacle may be a moving human, or should the robot plan a path around the object because it may be a cardboard box? With sonar alone, these states are aliased and differentiation is impossible. The problem posed to navigation because of sensor aliasing is that, even with noise-free sensors, the amount of information is generally insufficient to identify the robot’s position from a single-percept reading. Thus techniques must be employed by the robot programmer that base the robot’s localization on a series of readings and, thus, sufficient information to recover the robot’s position over time.

5.2.3 Effectors Noise

The challenges of navigation do not lie with sensor technologies alone. Just as robot sensors are noisy, limiting the information content of the signal, so robot effectors are also noisy. In particular, a single action taken by a mobile robot may have several different possible results, even though the initial state before the action was taken is well known from the robot’s point
of view. In short, mobile robot effectors introduce uncertainty about future state. Therefore the simple act of moving tends to increase the uncertainty of a mobile robot. There are, of course, exceptions. Using cognition, the motion can be carefully planned so as to minimize this effect, which may lead to a certainty state. Furthermore, when the robot’s actions are taken in concert with careful interpretation of sensory feedback, it can compensate for the uncertainty introduced by noisy actions using the information provided by the sensors.

First, however, it is important to understand the precise nature of the effector noise that impacts mobile robots. It is important to note that, the error in motion is viewed as an error in odometry by the robot to estimate its own position over time using knowledge of its kinematics and dynamics. The true source of error generally lies in an incomplete model of the environment. For instance, the robot does not model the fact that the floor may be sloped, the wheels may slip, and a human may push the robot. These unmodeled sources of error result in inaccuracy between the physical motion of the robot, the intended motion of the robot, and the proprioceptive sensor estimates of motion. In odometry (wheel sensors only) and dead reckoning (also heading sensors) the position update is based on proprioceptive sensors. The movement of the robot, sensed with wheel encoders or heading sensors or both, is integrated to compute position. Since the sensor measurement errors are integrated, the position error accumulates over time. Thus the position has to be updated from time to time by other localization mechanisms. Otherwise the robot is not able to maintain a meaningful position estimate in the long run.

There are many sources of odometric error, from environmental factors to resolution:
• Limited resolution during integration (time increments, measurement resolution, etc.);
• Misalignment of the wheels (deterministic);
• Uncertainty in the wheel diameter and in particular unequal wheel diameter (deterministic);
• Variation in the contact point of the wheel;
• Unequal floor contact (slipping, nonplanar surface, etc.).

Some of the errors might be deterministic (systematic), thus they can be eliminated by proper calibration of the system. However, there are still a number of nondeterministic (random) errors which remain, leading to uncertainties in position estimation over time.

From a geometric point of view, one can classify the errors into three types:

1. Range error: integrated path length (distance) of the robot’s movement \( \rightarrow \) sum of the wheel movements
2. Turn error: similar to range error, but for turns \( \rightarrow \) difference of the wheel motions
3. Drift error: difference in the error of the wheels leads to an error in the robot’s angular Orientation.

5.3 ARCHITECTURE FOR NAVIGATION

There are two types of architectures for mobile robot navigation. A behavior-based system (Figure 5.3) may have multiple active behaviors at any one time. Even when individual behaviors are tuned to optimize performance, this fusion and rapid switching between multiple behaviors can
negate that fine-tuning. Often, the addition of each new incremental behavior forces the robot designer has to retune all the existing behaviors to ensure that the new interactions with the freshly introduced behavior are all stable.

**Figure 5.3 Architecture for behavior based navigation**

![Diagram of behavior based navigation](image1)

**Figure 5.4 Architecture for map based navigation**

![Diagram of map based navigation](image2)

In contrast to the behavior-based approach, the map-based approach includes both localization and cognition modules (Figure 5.4). In map-based navigation, the robot explicitly attempts to localize by collecting sensor data, then updating some belief about its position with respect to a map of the environment. The key advantages of the map-based approach for navigation are as follows:
• The explicit, map-based concept of position makes the system’s belief about position transparently available to the human operators. The existence of the map itself represents a medium for communication between human and robot, the human can simply give the robot a new map if the robot goes to a new environment.

5.4 PATH PLANNING AND OBSTACLE AVOIDANCE

5.4.1 Path Planning

Motion planning for robots is basically the problem of given an initial and a final configuration of the robot in its workspace, find a path, starting at the initial configuration and terminating at the goal configuration, while avoiding collisions with the obstacles. From a theoretical point of view, motion planning has already been solved, in practice the complexity of algorithms still has many problems. In many practical problems path planning is not difficult. This is the reason why emphasis has been currently placed on designing efficient algorithms that solve problems that may not be complete. The robots must follow certain trajectories in order to complete their operations without colliding with other robots. In general they will be doing repetitive operations whose paths are determined. But, if for any reason the trajectory has to be changed, the robots must plan their trajectories on line.

5.4.2 Road Map Path Planning

Road map approaches capture the connectivity of the robot’s free space in a network of 1D curves or lines, called road maps. Once a road map is constructed, it is used as a network of road (path) segments for robot motion planning. Path planning is thus reduced to connecting the initial and goal positions of the robot to the road network, then searching for a series of
roads from the initial robot position to its goal position. The road map is a
decomposition of the robot’s configuration space based specifically on
obstacle geometry. The challenge is to construct a set of roads that together
enable the robot to go anywhere in its free space, while minimizing the
number of total roads. Generally, completeness is preserved in such
decompositions as long as the true degrees of freedom of the robot have been
captured with appropriate fidelity. We describe two road map approaches
below that achieve this result with dramatically different types of roads. In the
case of the visibility graph, roads come as close as possible to obstacles and
resulting paths are minimum-length solutions. In the case of the Voronoi
diagram, roads stay as far away as possible from obstacles.

![Figure 5.5 Visibility graph](image1) ![Figure 5.6 Verona diagram](image2)

### 5.4.2.1 Visibility Graph

The visibility graph for a polygonal configuration space consists of
edges joining all pairs of vertices that can see each other (including both the
initial and goal positions as vertices as well). The unobstructed straight lines
(roads) joining those vertices are obviously the shortest distances between
them. The task of the path planner is thus to find the shortest path from the
initial position to the goal position along the roads defined by the visibility graph (Figure 5.5).

### 5.4.2.2 Voronoi Diagram

Contrasting with the visibility graph approach, a Voronoi diagram is a complete road map method that tends to maximize the distance between the robot and obstacles in the map. For each point in the free space, compute its distance to the nearest obstacle. Plot that distance in (Figure 5.6) as a height coming out of the page. The height increases as you move away from an obstacle. At points that are equidistant from two or more obstacles, such a distance plot has sharp ridges. The Voronoi diagram consists of the edges formed by these sharp ridge points.

### 5.5 Potential Field Path Planning

Potential field path planning creates a field, or gradient, across the robot’s map that directs the robot to the goal position from multiple prior positions. This approach was originally invented for robot manipulator path planning and is used often and under many variants in the mobile robotics community. The potential field method treats the robot as a point under the influence of an artificial potential field. The robot moves by following the field, just as a ball would roll downhill. The goal (a minimum in this space) acts as an attractive force on the robot and the obstacles act as peaks, or repulsive forces. The superposition of all forces is applied to the robot, which, in most cases, is assumed to be a point in the configuration space in Figure 5.7. Such an artificial potential field smoothly guides the robot toward the goal while simultaneously avoiding known obstacles.
5.6 LOCAL OBSTACLE AVOIDANCE

Local obstacle avoidance focuses on changing the robot’s trajectory as informed by its sensors during robot motion. The resulting robot motion is both a function of the robot’s current or recent sensor readings and its goal position and relative location to the goal position. The obstacle avoidance algorithms presented below depend on varying degrees of the existence of a global map and on the robot’s precise knowledge of its location relative to the map. Despite their differences, the algorithms below can be termed as obstacle avoidance algorithms because the robot’s local sensor readings play an important role in the robot’s future trajectory. We first present the simplest obstacle avoidance systems that are used successfully in mobile robotics. The Bug algorithm represents such a technique in that only the most recent robot sensor values are used, and the robot needs, in addition to current sensor values, only approximate information regarding the direction of the goal.
5.6.1 Tangent Bug Algorithm

Tangent Bug algorithm is selected because of its ease of implementation and deployment. The basic idea of Tangent Bug is given below:

- A motion-to-goal behavior as long as way is clear or there is a visible obstacle boundary point that decreases heuristic distance.
- A boundary following behavior invoked when heuristic distance increases.
- A value $d_{\text{followed}}$ which is the shortest distance between the sensed boundary and the goal.
- A value $d_{\text{reach}}$ which is the shortest distance between blocking obstacle and goal.
- Terminate boundary following behavior when $d_{\text{reach}} < d_{\text{followed}}$

The simulator program was written in such a way that, the user can select an arbitrary point on the environment. The robot then uses its sensory information and real-time distance measurements to execute the Tangent Bug algorithm to calculate the optimal path to the selected point. The simulator has options to include obstacles of different sizes and shapes to the environments. The algorithm is use for a robot equipped with a distance sensor.
The Tangent Bug algorithm is as given below

1. repeat
   a) Compute continuous range segments in view
   b) Move toward \( n \in \{T,O_i\} \) that minimizes \( h(x,n) = d(x,n)+d(n,g_{goal}) \)
      until
      a) goal is encountered, or
      b) the value of \( h(x,n) \) begins to increase
2. Follow boundary continuing in same direction as before repeating
   a) update \( \{O_i\} \), \( d_{reach} \) and \( d_{followed} \)
      until
      b) goal is reached
      c) a complete cycle is performed (goal is unreachable)
      d) \( d_{reach} < d_{followed} \)

Once the simulation is started, the robot starts moving towards the direction \( q_{goal} \). If the sensor feels the presence of obstacles, the distance function has some values less than the threshold; some intervals of continuity are defined on the boundary of the sensed objects.

If the direction between robot and intersects at an interval, the robot will now move in the direction of the end point \( O_i \) of some interval of continuity which minimizes the following heuristic function:

\[
d(x,n)+d(n,q_{goal})
\]

(5.1)

When the direction between robot and the goal does not intersect at an interval of continuity, the robot will start again to move in the goal direction. In order to prevent the failing of algorithm at local minima of the heuristic function, the function begins to increase a boundary.
In the beginning, the algorithm moves, the robot in the same direction as the most recent motion step. The boundary following motion ends when the following relationship is fulfilled:

\[ d_{\text{reach}} < d_{\text{followed}} \]  \hspace{1cm} (5.2)

and the robot will start again to move in the goal direction. In the above equation \( d_{\text{reach}} \) is the shortest distance between the goal and the blocking obstacle and \( d_{\text{followed}} \) is the shortest distance between the boundary which had been sensed and the goal. in this thesis, a function was implemented inside the simulator which calculates the distance \( d_{\text{reach}} \) and the motion direction. If the robot completes a cycle around the obstacle, goal will not be reached.

5.7 ANALYSIS OF EXPERIMENTAL RESULTS

The experimentation consisted in testing the path planning algorithms with regard to calculating the odometry values and finding the optimal path between the initial position and destination. To verify the effectiveness of the proposed method a series of experiments was conducted using Matlab and Webots simulator. The tangent bug algorithm has been applied to several maps using Webots simulator. The results obtained for each testing are as follows:

5.7.1 Experimental Results for Potential Field Algorithm

This method creates a field, or gradient, across the robot’s map that directs the robot to the goal position from multiple prior positions. The potential field method treats the robot as a point under the influence of an artificial potential field. The goal acts as an attractive force on the robot and the obstacles act as peaks, or repulsive forces. The superposition of all forces is applied to the robot, which, in most cases, is assumed to be a point in the
configuration space. Such an artificial potential field smoothly guides the robot toward the goal while simultaneously avoiding known obstacles. The path found from this method is shown in Figure 5.8.

Figure 5.8 Robot is finding the path
e) Robot trajectory created by the superposition of forces

(f) Robot path extracted to avoid obstacles and reach goal

Figure 5.8 (Continued)

5.8 POSE ESTIMATION

5.8.1 Pose Estimation Based on Particle Filters

Pose estimation are still some of the key challenges in the area of robotics navigation, the basic requirement for an autonomous mobile robot is its capability to elaborate the sensor measurements to localize itself with respect to a global reference points. For this purpose the odometric values or the sensor measurements have to be fused together by means of particle filters. Earlier particle filters were limited to low-dimensional estimation problems, such as robot localization in known environments. More recently, particle filters are used in spaces with as many as 100,000 dimensions. This thesis presents some of the recent innovations on the use of particle filters in robotics.

The adoption of probabilistic techniques has been one of the key developments in the field of robotics. Earlier researchers focused on path planning and control problems in fully deterministic robot environments (Khargonekar et al 1985). This changed radically in mid 1980s, when the
approaches such as Brook’s behavior-based architecture generated control directly in response to sensor measurements. Probabilistic robotics integrates imperfect models and imperfect sensors through probabilistic laws, such as Bayes rule.

This thesis work surveys some of the recent developments and points out some of the opportunities and pitfalls specific to robotic problem domain. Particle filters (Pitt and Shephard 1999), Braathen et al (2002) comprise a broad family of sequential Monte Carlo algorithms for approximate inference in partially observable Markov chains (Doucet et al 2000). Early successes of particle filter implementations can be found in the area of robot localization, in which the robot’s pose has to be recovered from sensor data. These advances have led to a critical increase in the robustness of mobile robots, and the localization problem is now widely considered to be solved. More recently, particle filters have been in the core of solutions to much higher dimensional robot problems. One of the particle filter algorithm known as FastSLAM has been demonstrated to solve problems with more than 100,000 dimensions in real-time. However, a range of problems still exist with probabilistic techniques. In robotics, all models lack important state variables that systematically affect sensors and actuator noise.

5.9 PARTICLE FILTERS

Particle filters are approximate techniques for calculating posteriors in partially observable controllable Markov chains with discrete time (Figure 5.9). Suppose the state of the Markov chain at time ‘k’ is given by ‘x_k’ where ‘x_k’ is the state vector describing the location and orientation of the robot. ‘u_k’ is the control vector at time ‘k-1’ to drive the robot to a state ‘x_k’ at time ‘k’, ‘m_i’ is a vector describing the location of the i^{th} landmark whose true location is assumed time invariant. ‘z_{ik}’ is an observation taken from the robot of the location of the i^{th} landmark at time ‘k’(Figure 5.10)
In addition, the following sets are also defined. 

- $X_{0:k} = \{x_0, x_1, \ldots, x_k\} = \{X_{0:k-1}, x_k\}$: the history of the robot’s locations
- $U_{0:k} = \{u_0, u_1, \ldots, u_k\} = \{U_{0:k-1}, u_k\}$: the history of control inputs
- $Z_{0:k} = \{z_0, z_1, \ldots, z_k\} = \{Z_{0:k-1}, z_k\}$: the set of all landmark observations

Figure 5.9  Schematic showing the principle of stochastic uncertainty propagation

Figure 5.10  (a) Localization with only odometric measures,  
(b) Localization with particle filters
5.10 PARTICLE FILTERS IN LOW DIMENSIONAL STATE SPACE

Mobile robot localization addresses the problem of estimation of a mobile robot’s pose relative to a given map from sensor measurements and controls. The pose is specified by a two-dimensional Cartesian coordinates and the robot’s orientation in space. More general is the global localization problem, which is the problem of localizing a robot under global uncertainty.

Particle filters are commonly known as Monte Carlo Localization (MCL) (Arulampalam et al 2002) (Liu and Chen 1998). MCL’s original development was motivated by the condensation algorithm (Bonci et al 2004). In most variants of the mobile robot localization problem, the particle filters have been consistently found to outperform alternative techniques, including parametric probabilistic techniques such as the Kalman filters (Julier and Uhlmann 1997) and more traditional techniques. MCL has been implemented with as few as 100 samples as shown in Figure 5.11. And convergence of landmark uncertainty is shown in Figure 5.12.

The Monte Carlo method is just one of the many methods for analyzing uncertainty propagation, where the goal is to determine how random variation, lack of knowledge, or error affects the sensitivity, performance, or reliability of the system that is being modeled. Monte Carlo is a sampling method because the inputs are randomly generated from probability distributions to simulate the process of sampling from an actual population. So, the distribution for the inputs that most closely matches data that are already available or the best represents the current state of knowledge.
Figure 5.11 (a) Distribution of observed standard deviations in the localization estimate, (b) Distribution of estimated standard deviations in the localization estimate

Figure 5.12 The convergence of landmark uncertainty

5.11 PARTICLE FILTERS IN HIGH DIMENSIONAL STATE SPACE

A further important aspect in the context of particle filter is the ability to deal with high-dimensional state spaces. For example, problems such as multi robot tracking SLAM and mapping involve high-dimensional state spaces. As the size of the state space typically grows exponentially in the
number of dimensions, appropriate techniques are needed to efficiently approximate the full posterior and update it accordingly.

Simultaneous localization and mapping problem (SLAM) as shown in Figure 5.13 addresses the problem of building a map of the environment with a moving robot. The absence of an initial map in the SLAM problem makes it impossible to localize the robot during mapping using algorithms like MCL. Furthermore, the robot faces a challenging data association problem as shown in Figure 5.14 of determining whether two environment features, observed at different point in time correspond to the same physical feature in the environment. The solution to the above stated problem was based on Extended Kalman Filters or EKFs. The use of Kalman filtering techniques requires deriving a stochastic state space representation of the robot model and the measure process. Formally this can be readily performed by applying the kinematic model of the robot and the knowledge of the measure equipment. The EKF techniques are based on some fixed values of the input and measurement noise covariance matrices. There are some limitations to the EKF algorithm.

- EKF algorithms can rarely manage more than a thousand features in realistic time
- EKFs cannot represent negative information because they may give rise to non-Gaussian posteriors, which cannot be represented by EKFs.
- The data association problem cannot be solved using EKFs.

Recent research has led to a family of so-called Rao-Blackwellized particle filters that in the context of SLAM, lead to solutions that are significantly more efficient than the EKF. These particle filters require time of $O(m \log n)$ instead of $O(n^2)$ where $M$ is the number of particles and $n$ is the number of features.
Figure 5.13 The structure of SLAM problem

Figure 5.14 Data association graph
This thesis has described some of the recent successes of particle filters in the field of robotics. Earlier, particle filters were applied to low-dimensional state spaces. Recently, particle filters have provided new solutions to challenging higher-dimensional problems such as the problem of multi-robot tracking. Nowadays particle filter approach is among the most efficient and scalable solutions for the self-localization problem of mobile robot in unknown and unstructured environments.

Despite this progress, there exist plenty of opportunities for future research. Appearance and pose-based SLAM methods offer a radically new paradigm for mapping and location estimation without the need for strong geometric landmark descriptions. The key challenge now is to implement SLAM solutions to problems where robotics can truly contribute like driving hundreds of kilometers under a forest canopy or mapping a whole city without using global positioning systems (GPS) and to demonstrate the autonomous location and mapping in planetary terrains. The demo model and performance analysis is available in Appendix 2.

5.12 CONCLUSION

This work includes the detailed study of different challenges faced when a mobile robot moves in an unknown and unstructured environment. The various path planning techniques were studied and tested. A detailed simulation work of the potential field path planning technique and tangent bug obstacle avoidance technique was implemented. A simulation of different maps were designed and tested using Webots Robotic Simulator. A working model of a robot finding its way through different unknown paths and avoiding obstacles to reach its goal were successfully simulated using Webots. A performance evaluation of the different odometric values generated as the robot moves through different environments were done and paths were generated from each environment that was created using the simulator and various graphs were plotted.