Collaborative Mapping Algorithm

Scientific Seminar

submitted by
Ee Heng Chen

NEUROSCIENTIFIC SYSTEM THEORY
Technische Universität München

Prof. Dr Jörg Conradt

Supervisor: Dipl.-Inf. Nicolai Waniek
Final Submission: 06.07.2016

In your final hardback copy, replace this page with the signed exercise sheet.

Contents

Chapter 1

Introduction

In the world today, robots are no longer a topic of science fiction. Leading robotic companies such as KUKA, ABB, and Staeubli are capable of manufacturing industrial robots that could carry out tasks with micrometer precision. As the ultimate goal of robotics is to bring robots into our daily lives, one of the key challenges is to build mobile robots that could navigate and avoid obstacles in different surroundings. For navigation, mobile robots require a map of the surrounding environment and their current position in it. This can be achieved by either providing an accurate map of the environment, or allowing access to an external reference system such as GPS. However, this is not always the case as GPS is not readily available for indoor environment and no maps are available for an unknown environment [GKSB10]. A way to overcome this is to allow robots to build the map of the surrounding and navigate based on it, a topic in the field of robotics known as Simultaneous Localization and Mapping (SLAM).

While there is a vast number of literature and research work dedicated to solve the SLAM problem in a Single Robot (SR) setting, there exists only a number of work that involves Multiple Robots (MR) setting. By involving MR, the mapping process could be done at a faster rate and possibly with less computational effort. However, the use of MR leads to additional issues such as connectivity, map merging and the building of a common map. The goal of this scientific seminar will be to review a few approaches for collaborative mapping in a MR setting, with the assumption that the robots have perfect connectivity between. First we will discuss about a few types of map representation used in various mapping algorithm in chapter 2. Then we will review related works on SLAM in chapter 3, followed by an introduction on two approaches to tackle SLAM: Rao-Blackwellized Particle Filter in chapter 4 and graph-based method in chapter 5. Additionally an approach for collaborative mapping without the SLAM formulation is introduced in chapter 6. Finally we will compare and discuss the three approaches together in chapter 7 and conclude the report in chapter 8.

Chapter 2

Map Representation

Maps built by robots during mapping are based on observations of the surrounding and the location where these observations are made. In general the maps built can be divided into two types [GKSB10] : landmark maps and dense representations. Landmark maps are built by determining the pose of landmarks in the surroundings. It is preferred when a camera is used or when locally distinguishable features of the surroundings can be easily identified. The landmarks are defined once in the map and methods to associate an observation to the corresponding landmark are required. Dense representations are maps that are built through scan matching of range sensor measurements. They show which part of the surrounding is blocked by an obstacle and which part is obstacle-free. This type of map includes point cloud representation and occupancy grid map.

When compared together, the two types of map representation have different benefits and drawbacks. For landmark maps, only the pose of the landmarks is saved in the map, making the memory complexity dependent on the number of landmarks and not the total time. Once a landmark is saved, the robots have to be able to correctly identify it if the same landmark is observed again. This is done through data association of the observed landmark. In comparison, dense representations do not carry out data association on landmarks but rather through scan matching that tries to align a locally built map to a global map. Here the landmarks do not need to be explicitly specified. However, to carry out scan matching, all the measured observation needs to be saved in the map, causing the memory complexity to be high for large maps.

Chapter 3

Related Works on SLAM

In the field of robotics, SLAM is a well-known mobile robot mapping problem. The SLAM problem involves two main parts; estimating the position of a robot relative to a map, and generating a map using sensor measurements and the estimated position [HBFT03]. Mathematically, the SLAM problem can be formulated probabilistically, in which the goal it to estimate the joint posterior density distribution of the robot's pose x and its map m given the recorded observations from the robot's sensor z, the odometry measurement of the robot u and the initial pose of the robot x_0;

$$p(x, m|z, u, x_0) \tag{3.1}$$

The probabilistic formulation is based on the Bayesian Belief Network (see figure 3.1) where the robot's pose and the map are unknown variables that can be determined using the measurements and observations made. A detailed description of the SLAM problem and its formulation can be found in [DWB06] and will therefore not be mentioned here.

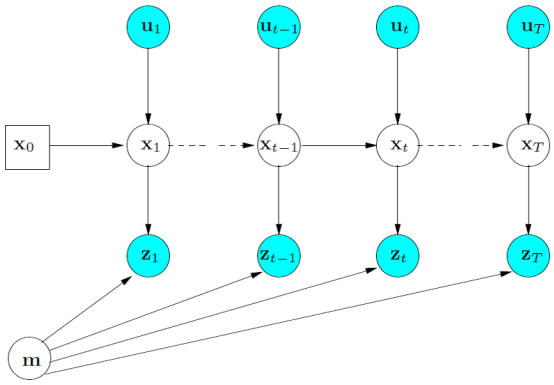

Figure 3.1: Bayesian Belief Network for SLAM [GKSB10]

To solve the SLAM problem, many methods and approaches have been proposed. As mentioned in [GKSB10] these methods can be generally divided into two cate-

gories; filtering and smoothing. Filtering models SLAM as an online state estimation problem that updates as new measurements become available. The state here refers to the map and the robot's current position in the map. Some of the methods that utilize filtering include the use of Kalman Filter (KF) [DWB06] [SSC90] [M+99], Extended Kalman Filter (EKF) [DWB06], Rao-Blackwellized Particle Filter (RBPF) [GTS+07] [DDFMR00], and FastSLAM [GKSB10] [MTK+02]. As mentioned in [DWB06], the use of KF and EKF are both very well-known but suffer some key problems. They both need to update the entire covariance matrix during the update step which leads to high computational effort. Furthermore, the non-linearity of the process model could cause the EKF to produce inconsistent solutions. In comparison, RBPF is able to represent a non-linear process model directly and at the same time reduce sample space by partitioning the joint state space distribution [DWB06]. An extension to RBPF is the FastSLAM algorithm that uses EKF to update the individual landmark based maps in each particle [MTK+02].

On the other hand, smoothing tries to tackle the so-called "full SLAM problem". The idea is to estimate the full robot trajectory using the entire measurement dataset and with the help of error minimization techniques. Typically the SLAM problem is represented as graphs for smoothing. GraphSLAM [TM06], Tectonic-SAM [NSD07], C-SAM [AN08] and ISAM [KRD08] are methods that use smoothing with graph representation. GraphSLAM employs variable elimination techniques to reduce the dimensionality. In Tectonic-SAM, sub maps are generated first before being merged into a common map. Merging is done by aligning the individual sub maps and optimizing the "separators" that separate the sub maps. The C-SAM method is based on merging featured based landmark maps and operates as a batch algorithm [KKF+10]. Unlike the other methods, ISAM is an online algorithm that uses variable reordering for sparse factorization. It has the advantage that it is able to be run recursively in real time.

All of the methods discussed until now are to tackle the single-robot SLAM problem. In chapters 4 and 5, we will first discuss more in depth about RBPF and ISAM respectively. Then we will discuss about the approaches taken to extend them for MR setting. In addition to SLAM, we will look into a mapping approach using generalized polylines which do not uses the probabilistic model of SLAM. Finally we will evaluate and compare the different approaches discussed.

Chapter 4

Rao-Blackwellized Particle Filter

In this chapter, we will review the RBPF approach to solve the SLAM problem. A detailed overview of the RBPF algorithm for a SR setting will be discussed first, followed by an extension for a MR setting. The overview will include the mathematical formulation for the RBPF, while the idea of "anchor nodes" is introduced in the extension for the merging of maps from multiple robots. The approaches discussed here are based on the works of [HBFT03] and [How06]. The reader is assumed to have basic understanding about PF to understand the terms used for RBPF.

4.1 Single Robot Setting

The high dimension sample-space of the SLAM problem formulated in equation 3.1 causes the application of the standard PF to be computational infeasible. However the sample space dimension can be reduced by a technique called Rao-Blackwellization. This technique partitions the joint state-space of SLAM into two components:

$$p(x_{1:t}, m|z_{1:t}, u_{0:t-1}, x_0) = p(m|x_{1:t}, z_{1:t}, u_{0:t-1}, x_0)p(x_{1:t}|z_{1:t}, u_{0:t-1}, x_0) \tag{4.1}$$

where the first component is the distribution of the map generated, while the second component is the distribution of the trajectory of the robot, also known as proposal distribution. The advantage of this formulation lies in the ability to calculate the first term analytically with the trajectory estimated from the second term.

The RBPF is constructed by creating multiple particles that contain the tuple $< x_t^{(i)}, m_t^{(i)}, w_t^{(i)} >$. Here $x_t^{(i)}$ is the pose of the robot at time t in particle i, $m_t^{(i)}$ is the map generated conditioned on particle i, and $w_t^{(i)}$ is the weight of the particle. The filter is updated as follows:

$$x_t^{(i)} = A(u_{t-1}, x_{t-1}^{(i)}) \tag{4.2}$$

$$m_t^{(i)} = M(z_t, x_t^{(i)}) + m_{t-1}^{(i)} \tag{4.3}$$

$$w_t^{(i)} = S(z_t, x_t^{(i)}, m_{t-1}^{(i)}) w_{t-1}^{(i)} \tag{4.4}$$

where A is the action model that gives a random pose from the proposal distribution, M is an incremental map generator that utilizes simple ray-tracing algorithm to create partial occupancy grid map, and S is the sensor model . Through resampling, the weight of the particles with consistent map is increased while the particles with lower weight will eventually be discarded.

4.2 Multiple Robots Setting

The above mentioned SR RBPF can be generalized to include MR. If the initial poses are known, then equation 4.1 can be easily extended to estimate the posterior over multiple robots and one map from which the pose of the robots are based on:

$$\begin{aligned} &p(x_{1:t}^1, x_{1:t}^2, m | z_{1:t}^1, u_{0:t-1}^1, x_0^1, z_{1:t}^2, u_{0:t-1}^2, x_0^2) = \\ &\quad p(m | x_{1:t}^1, z_{1:t}^1, x_0^1, x_{1:t}^2, z_{1:t}^2, x_0^2) \\ &\quad p(x_{1:t}^1 | z_{1:t}^1, u_{0:t-1}^1, x_0^1) p(x_{1:t}^2 | z_{1:t}^2, u_{0:t-1}^2, x_0^2) \end{aligned} \tag{4.5}$$

where there are two distribution term for the trajectory of two robots instead of one in equation 4.1. The term $u_{0:t-1}$ is dropped in the distribution of map generated for simplicity. Here, the number of distribution term can be expanded to include n number of robots. However this formulation is based on the assumption that the trajectory and observation of one robot is independent from the others. This means that the robots are assumed to be outside of each other's sensor range and that they exclude the observations made by observing other robots.

For the case where the initial poses of robots are unknown, especially if the robots start far apart, the RBPF can still be applied by determining the relative poses of robots during an encounter. Initially, the RBPF is initialized for robot 1, which has an arbitrary initial pose. After the first encounter with robot 2 at time $t = s$, the measurement data from robot 2 will be incorporated into the map of robot 1:

$$\begin{aligned} &p(x_{1:t}^1, x_{s:t}^2, m | z_{1:t}^1, u_{0:t-1}^1, x_0^1, z_{s:t}^2, u_{s:t-1}^2, \Delta_s^2) = \\ &\quad p(m | x_{1:t}^1, z_{1:t}^1, x_0^1, x_{s:t}^2, z_{s:t}^2, x_s^1, \Delta_s^{21}) \\ &\quad p(x_{1:t}^1 | z_{1:t}^1, u_{0:t-1}^1, x_0^1) p(x_{s:t}^2 | z_{s:t}^2, u_{s:t-1}^2, x_s^1, \Delta_s^{21}) \end{aligned} \tag{4.6}$$

where delta is the relative pose between robot 1 and 2 at time s. Δ is used to transform the trajectory from robot 2 to robot 1, effectively creating a global map conditioned on robot 1. This approach ignores all subsequent encounters after the first.

As the data of robots are incorporated after the first encounter, the information of the robots prior to the encounter is lost. One way to utilize the lost

information is to create "virtual robots" that travel back in time along the same trajectory. This means that after the first encounter, robot 2, will have the trajectory $x_{s:t}$ while the virtual robot 2, also known as acasual instance, has trajectory $x_{1:s}$ (see figure 4.1). Updates for both robots are done at the same rate and are initialized per-particle at the first encounter. However, this approach causes latency. If the latency is to be removed, the filter needs to be updated with each observation during an encounter, causing 2 drawbacks; the RBPF slows down and the resampling caused by multiple filter updates tends to impoverish the sample set in the vicinity of remaining robots.

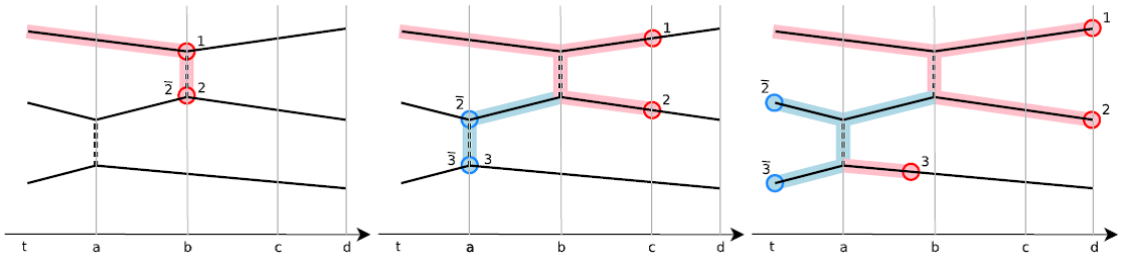

Figure 4.1: Encounters for between robots. Here the path highlighted in blue is the trajectory that goes backwards in time while the path in red is the trajectory that follows the time. After the first encounter between robot 1 and robot 2, a virtual robot $\bar{2}$ is created that follows the past trajectory of robot 2. This is repeated again when robot $\bar{2}$ encounters robot 3. However since the second encounter happened as robot $\hat{2}$ travels back in time, this causes a latency of $2(b-a)s$ between robot 3, and robots 1 and 2 [How06]

Chapter 5

Pose Graph

The following chapter tackles the SLAM problem using the graph-based solution of ISAM. First we will discuss an overview of graph-based SLAM [GKSB10], followed by the online approach introduced in ISAM [KRD08]. The approach uses partial variable reordering to compute a sparse factorization matrix. An extension of ISAM to include MR setting will be discussed, which is based on the work of [KKF+10]. For this chapter, the reader is assumed to have basic knowledge of graph theory, linear algebra, and error minimization techniques.

5.1 Graph-based SLAM

The graph-based approach of solving the SLAM problem involves building a pose-graph. The construction of the graph can be divided into two main parts; front end which builds the graph by interpreting sensor data, and back end which deals with graph optimization.

At the front end, a graph is constructed by adding nodes and edges. Each node represents the robot's pose or a landmark at a point in time. The edges are the constraints enforced onto the two nodes that it connects, showing how a pose is related to another pose or to a landmark. Also encoded in the nodes are the measurements made by the robot in that pose. Generally the constraints represented by the edges are based on either aligning the observation made by the two connected nodes or estimating the next pose based on the odometry measurements.

In the back end, the graph created in the front end is optimized. Assuming that the measurement model is Gaussian, for a certain configuration for nodes I and j, the optimization is done by finding the configuration for the nodes that maximizes the likelihood of the measurements (constraint optimization). This includes calculating an error function e and minimizing the negative log likelihood F:

$$e_{ij}(x_i, x_j) = z_{ij} - \hat{z}_{ij}(x_i, x_j) \tag{5.1}$$

$$F(x) = \sum_{<i,j>\in C} e_{ij}^T \Omega_{ij} e_{ij} \tag{5.2}$$

where $e_{ij}(x_i, x_j)$ represents the constraint that tries to align the observation made at nodes i and j.

Given that an initial pose x_0 is known, a numerical solution for the minimization of $F(x)$ can be computed by applying first order Taylor expansion for local approximation, which is the basis for optimization methods such as Gauss-Newton or the Levenberg-Marquadt algorithms. As derived in [GKSB10], the use of these algorithms leads to the linear system:

$$H\Delta x^* = -b \tag{5.3}$$

where H is the information matrix (Hessian matrix) of the system and has non zeros between poses connected by a constraint. H is a sparse matrix with a block like structure. The blocks in H are symmetric and thus an upper triangle computation is sufficient. It is important to note that $F(x)$ is invariant under rigid transformation, due to the error function dealing only with the relative position between two poses. Hence one of the increments needs to be constrained to zero so that the solution to equation 5.3 is not underdetermined.

5.2 ISAM

The graph-based SLAM discussed in the previous sub-chapter can be reformulated to support real-time implementation, leading to a method called Incremental Smoothing and Mapping ISAM. ISAM is based on the formulation of the SLAM problem as a smoothing problem which still represents the Bayesian Belief Network in figure 3.1.

In comparison to filtering that updates the map from one step pose prediction, smoothing tries to estimate the robot's complete trajectory. The smoothing approach can be formulated as the joint probability of all the variables and measurements:

$$P(X, L, U, Z) \propto P(x_0) \prod_{i=1}^{M} P(x_i|x_{i-1}, u_i) \prod_{k=1}^{K} P(z_k|x_{i_k}, l_{j_k}) \tag{5.4}$$

where the first term is the prior on the initial state, the second term is the motion model (process model) and the third term the landmark measurement model. The two models are assumed to be Gaussian:

$$x_i = f_i(x_{i-1}, u_i) + w_i \tag{5.5}$$

$$z_K = h_k(x_{i_k}, l_{j_k}) + v_k \tag{5.6}$$

where w and v are normally distributed zero-mean noise.

Since the goal of SLAM is to maximize the joint posterior density distribution of the robot's trajectory and its map, the MAP estimate can be obtained by calculating the error function and minimizing the negative log-likelihood using the two models mentioned above:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} ||A\theta - b||^2 \tag{5.7}$$

where theta consist of all the pose and landmark variables δx and δl, A is a sparse measurement Jacobian matrix, and b is the prediction error. In comparison to equation 5.1, here the odometry measurements are considered in addition to the landmark observations. The covariance matrix is dropped from the equation as it can be eliminated by pre-multiplying A with the inverse square root of the covariance matrix [KRD08].

Instead of applying non-linear optimization methods to obtain a solution to equation 5.7 as shown in the previous sub-chapter, the minimization problem is solved using QR factorization. By applying QR factorization, A can be decomposed into an orthogonal Q and a upper triangle square root information matrix R and equation 5.7 can be simplified to:

$$R\theta^* = d \tag{5.8}$$

where d is the multiplication of b with Q. Since R is an upper triangular matrix, θ can be computed through back substitution, thus resulting in the estimate of the robot's complete trajectory and the map, conditioned on all the measurements. The QR factorization is done by using Givens rotations to zero out the elements below the diagonal of A, effectively forming the matrix R. If the Givens rotations are determined using the method described in [GVL12], it is numerically stable and accurate.

The algorithm can be implemented in real-time by adding new rows of measurements to R and d during each update. The new R is augmented with columns of zeroes to maintain the square matrix form. The upper triangular matrix structure of this new R can be reformed by using Givens rotations to zero out the elements that are below the diagonal in the newly added rows. At the same time d is updated with the same Givens rotations. Since R is sparse, only a constant number of Givens rotations are needed. The number is independent from the size of the trajectory and the map.

In the case of trajectory loops, R can become dense as loops introduce correlation between current pose and previously explored poses. It leads to “fill-in”,

which is the addition of non-zero entries beyond the sparsity pattern of matrix R. This is avoidable by using variable reordering, a method where the number of entries in R can be reduced by reordering the columns of R before applying the factorization. Here the heuristic COLAMD [DGLN04] is used to do variable reordering. As variable reordering slows down the algorithm, it is done periodically.

Since the approach builds a landmark map, the problem of data-association needs to be addressed. ISAM tackles this problem by using the maximum likelihood solution using Mahalanobis distance to include uncertainties between the robot's pose and the landmark. The solution can be reduced to a minimum cost assignment problem that is solved with the JVC algorithm [JV87]. In [KRD08] three ways of calculating the covariance for the Mahalanobis distance are introduced:

- Exact Covariance - computes the entire covariance matrix from R.
- Marginal Covariance - computes the covariance based on the entries in R that are required.
- Conservative Estimates - assumes that the covariance does not increase with new measurements

5.3 Multiple Relative Pose Graph

The SR ISAM approach discussed in the previous sub-chapter can be extended into a MR setting [KKF+10]. During an encounter, poses from the two robots can be connected together as a new constraint. This constraint can be based on the robots observing each other or a common environment (detected using match scanning). With this new constraint, the joint probability in equation 5.4 is expanded to include multiple robots:

$$P(X, L, C) \propto \prod_{r=0}^{R-1} \left(P(x_i|x_{i-1}, u_i) \right) \prod_{j=1}^{N} P(x_{i_j}^{r_j} | x_{i'_j}^{r'_j}, c_j) \tag{5.9}$$

where c is the encounter measurement. Here the third term in equation 5.4 for generating the map is not included. Again using the approach discussed in the previous sub-chapter, it is possible to get the posterior estimate of X and L for the SLAM problem, conditioned on all the measurements.

Since the robots build their own graphs in a local frame instead of a global frame, there exists a gauge freedom on the second robot trajectory, prior to the first encounter. This requires the second trajectory to be fixed with a prior. However before the initial encounter, the second prior cannot be well estimated. Fixing and removing the second prior after the first encounter is also problematic as it can cause the estimation to be far from a good linearization point. This can be

solved with the use of "anchor" nodes Δ as shown in figure 5.1. An anchor node is used to transform the robot's pose from local frame to a global frame, allowing the comparison of measurement to be done. This means that the pose graph of each individual robot does not need to be adjusted beforehand. For the anchor nodes, the minimization in equation 5.7 is expanded to include the following measurement model for encounters:

$$x_{i_j}^{r_j} = h_j^{'}(x_{i_j^{'}}^{r_j^{'}}, c_j, \Delta^{r_j}, \Delta^{r_j^{'}}) + \Gamma_j \tag{5.10}$$

where Γ is a zero-mean Gaussian noise. The anchor node also suffers from a gauge freedom problem which can again be solved by attaching a prior to the first anchor. The measurement constraint for loop was not considered in [KKF+10].

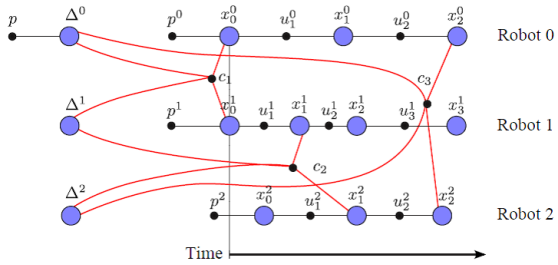

Figure 5.1: Relative pose graph constructed for multiple robots. All the encounters c are connected to an anchor node Δ that fixes the individual pose graphs to a common map. The first anchor node Δ is given a prior p due to the gauge freedom problem. [KKF+10]

Chapter 6

Mapping with Polylines

Moving away from the probabilistic approach of SLAM, [LLW05] proposes an approach of carrying out incremental mapping using sophisticated geometric handling of landmarks to merge locally built maps. The motivation of this approach is based on the conclusion made in [KFL+03]. It concludes that scan matching of the range data used to build local maps is not sufficient to merge maps into a global one. However, the use of features such as door handles and hallway junctions in an indoor environment allowed the merging to be done with a higher accuracy.

In [LLW05], the landmarks of a map are represented using generalized polylines with significant shape. Generalized polylines are a set of line segments, having specific ordering, where the vertices may or may not be connected, which is different from classical polyline (polygonal curve) where the vertices must be connected. In comparison to the standard 2D point representation of landmarks as x, y, and θ, the use of generalized polylines allow for the analysis of shape similarity, which greatly increases the possibility of correctly identifying the same landmarks in different maps. As the number of generalized polylines needed to represent the landmark is small, the dimensionality of the map can be reduced. For a SR setting, the incremental mapping algorithm can be summarized as follows:

1. Input data are processed to create generalized polylines.

2. The lines are grouped into segments and saved in a list. A separate list is created if the segments are too far apart.

3. Correspondence between scan (sensor data), S_t, and the map G_{t-1} is determined using shape similarity. The shape similarity does not depend on the position of the line in the map.

4. S_t is transformed by finding the minimum distance between the correspondences found. The Iterative Closest Point (ICP) algorithm is used to find the minimum distance.

5. After the transformation, S_t is merged into G_{t-1}, where polylines that overlap each other are removed and represented by just one line.

6. Similar line segments are merged iteratively to form a longer one by calculating the difference between polylines P_1 and P_2:

$$s(P_1, P_2) = \int_0^1 \left(T(P_1)(t) - T(P_2)(t) + \Theta(P_1, P_2) \right)^2 dt \tag{6.1}$$

$$\Theta(P_1, P_2) = \int_0^1 \left(T(P_1)(s) T(P_2)(s) \right)^2 ds \tag{6.2}$$

where T is the mapping to tangent space, and Θ is the term that takes into account the different orientation of the polylines. This merging is based on perceptual grouping that includes parallelism, collinearity, proximity of endpoints. Also to reduce the distortion error of the newly merged polyline, a direction histogram is used to find the main direction.

The above algorithm can be extended readily into a MRt setting. Here, each individual robots carry out mapping locally and the individual maps are merged together to form a common map by comparing two local maps. The main issue in this approach lies in the real-time implementation to merge correspondent maps. Here an approach is introduced where merging of the maps is done by determining the most salient object and finding a hypothetic matching counterpart through shape similarity. The idea is based on the assumption that the probability of wrong correspondence decreases as the complexity of the polyline's shape increases. The algorithm for map merging using this approach is as follows:

1. The Shape Complexity (SC) of an object in the map calculated:

$$sc(P) = s(P, L_p) \tag{6.3}$$

where L_p is a straight line with the same arc length as P. The object with the highest SC and with a value above a threshold is selected.

2. The corresponding object with maximum similarity is determined.

3. Based on shape similarity, two maps are aligned based on the object correspondence.

4. A position similarity measure such as Hausdorff Distance is applied to verify the correctness of the alignment. If the measure do not lie above a threshold, the alignment is discarded and the entire algorithm is repeated.

Chapter 7

Discussion

After introducing the three main methods to achieve MR mapping, the question arises as to which method offers the most practical advantages. Here in this chapter, these different methods will be evaluated and compared. As a reminder, it is assumed that the robots do not suffer from any connectivity issues during data transfer. For the sake of simplicity, the following notations will be used for this chapter: $M1$ is the RBPF algorithm, $M2$ is the pose graph approach, and $M3$ is the mapping approach using polylines.

With regards to complexity, $M1$ is independent of time and depends only on the number of particles used $O(n)$. Similar to the standard PF design, the issue of the number of particles required needs to be addressed. A tradeoff between high computational effort due to too many particles and low convergence due to small number of particles needs to be found. In [GTS+07] an approach was proposed to speed up the computation by adapting the proposal distribution used to calculate the weights, which was not tested on a MR setting. As for $M2$, the complexity is affected heavily by the back-substitution process. It is dependent on time, and has a quadratic time complexity if the matrix for the process is dense. However since matrix R is sparse and has a relatively constant entries per column (band-diagonal during exploration), the back-substitution process of R requires only $O(n)$ time [KRD08]. This statement still holds for loops and encounters as matrix R stays sparse after variable reordering. In [KRD08] it is shown that performing variable reordering periodically can reduce the computation time while still achieving the desired results. Another factor that affects the computational effort in $M2$ is the number of Givens rotations needed to formulate the upper triangular structure of R. Again as R is sparse, the number of Givens rotations needed is constant and is independent of the size of the trajectory or map.

When compared side by side, $M1$ and $M2$ both show a certain amount of advantage and limitation in their approach over the other. Given a fixed trajectory length, the dimensionality of the state space in $M1$ can be reduced by using more

robots, whereas $M2$ do not due to new measurements being added to the same information matrix. $M1$ assumes that the robots are able to determine their relative pose with low uncertainty during an encounter, while $M2$ sees an encounter as another measurement. The resampling step in $M1$ may cause particle impoverishment in the region, where the robot stops or encounters a loop, leading to low convergence. This is harder to tackle compared to using variable reordering to overcome the "fill-in" problem due to loops in $M2$ (see sub-chapter 5.2). In [TL05] the importance of initialization during the first encounter is stressed. While $M1$ makes the strong assumption that only the relative pose of the robots during the first encounter is accurate, $M2$ tackles the problem by using anchor nodes that "fixes" the encounter measurements to a global frame. The use of "virtual robots" for reverse time update in $M1$ allows data from different robots to be passed along in the filter, resulting in the situation where data is passed between robots that have yet to meet. For the building of landmark maps, $M2$ shows a clear advantage as the covariance matrix needed for data association can be calculated from the information matrix. Although the formulation in $M1$ builds an occupancy grid map, it can be modified to build a landmark map by using EKF for the map update step. This approach is also known as FastSLAM [MTK$^+$02].

In comparison to both $M1$ and $M2$, $M3$ do not make any assumption or deal with any uncertainty in measurement data that is incorporated into the probabilistic model of SLAM. The uncertainty can cause deviation error that will only be known when loop closing is done. The probabilistic approach of SLAM assumes that the robot's true pose is in the distribution. This means the approach might fail if discontinuity occurs such as the kidnapped robot problem. Furthermore, by considering polygonal lines instead of just reflection points as is the case for $M1$, the performance of scan matching used to build the map can be improved.

Chapter 8

Conclusion

In this scientific seminar, the following three approaches for MR collaborative mapping are introduced and evaluated:

- Rao-Blackwellized Particle Filter
- Graph-based SLAM
- Generalized polylines

These approaches were initially developed for SR setting before being extended into a MR setting. The first two approaches tackled the mapping problem based on the probabilistic formulation of SLAM. RBPF addresses the SLAM problem as a filtering problem, where a map is estimated and updated based on the current pose and the observation made. By making the assumption that time-reversed updates are possible, the trajectory information prior to an encounter in a MR setting can be used to build a "virtual robot" that travels back in time. However this approach causes latency in the filter update if an encounter involves a "virtual robot". On the other hand, the graph-based approach formulates the SLAM problem as a smoothing problem and solves it using error minimization techniques. For a MR setting, each robot creates its own graph and the graphs are combined by using anchor nodes that fixes the graphs together through encounters. An online graph-based approach known as ISAM was discussed that utilizes variable reordering method to keep the information matrix sparse, thus reducing the computational complexity. Finally, the third approach performs mapping through similarity measure of polylines. By aligning the polylines between two locally created maps, the maps can be merged to form a common map.

Although these approaches identified the steps required to build a global map collaboratively, there are still some open issues that are not discussed in this scientific seminar. One of the issues is connectivity. Here the robots are assumed to have perfect connectivity, which can be easily be affected by weak signal or system communication failures. Another issue is the need to develop efficient methods

to carry out exploration. This is to allow a more equal distribution of the area that each robot needs to map. Also, the idea of distributed data processing could help tremendously to ensure that mapping continues even when one of the robots malfunctions. These are a few open issues that arise only when mapping is carried out in a MR setting, and are potential topics for future work.

List of Figures

Abbreviation

EKF	Extended Kalman Filter
GPS	Global Positioning System
ICP	Iterative Closest Point
ISAM	Incremental Smoothing and Mapping
KF	Kalman Filter
MAP	Maximum a Posteriori Probability
MR	Multiple Robots
PF	Particle Filter
RBPF	Rao-Blackwellized Particle Filter
SLAM	Simultaneous Localization and Mapping
SC	Shape Complexity
SR	Single Robot

Bibliography

[AN08] Lars AA Andersson and Jonas Nygards. C-sam: Multi-robot slam using square root information smoothing. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2798–2805. IEEE, 2008.

[DDFMR00] Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[DGLN04] Timothy A Davis, John R Gilbert, Stefan I Larimore, and Esmond G Ng. Algorithm 836: Colamd, a column approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software (TOMS)*, 30(3):377–380, 2004.

[DWB06] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. 2006.

[GKSB10] Giorgio Grisetti, Rainer Kummerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[GTS+07] Giorgio Grisetti, Gian Diego Tipaldi, Cyrill Stachniss, Wolfram Burgard, and Daniele Nardi. Fast and accurate slam with rao–blackwellized particle filters. *Robotics and Autonomous Systems*, 55(1):30–38, 2007.

[GVL12] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[HBFT03] Dirk Hahnel, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 206–211. IEEE, 2003.

[How06] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, 2006.

[JV87] Roy Jonker and Anton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987.

[KFL+03] Kurt Konolige, Dieter Fox, Benson Limketkai, Jonathan Ko, and Benjamin Stewart. Map merging for distributed robot navigation. In *IROS*, pages 212–217, 2003.

[KKF+10] Been Kim, Michael Kaess, Luke Fletcher, John Leonard, Abraham Bachrach, Nicholas Roy, and Seth Teller. Multiple relative pose graphs for robust cooperative mapping. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3185–3192. IEEE, 2010.

[KRD08] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. isam: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[LLW05] Rolf Lakaemper, Longin Jan Latecki, and Diedrich Wolter. Incremental multi-robot mapping. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3846–3851. IEEE, 2005.

[M+99] Kevin P Murphy et al. Bayesian map learning in dynamic environments. In *NIPS*, pages 1015–1021, 1999.

[MTK+02] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Aaai/iaai*, pages 593–598, 2002.

[NSD07] Kai Ni, Drew Steedly, and Frank Dellaert. Tectonic sam: Exact, out-of-core, submap-based slam. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1678–1685. IEEE, 2007.

[SSC90] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles*, pages 167–193. Springer, 1990.

[TL05] Sebastian Thrun and Yufeng Liu. Multi-robot slam with sparse extended information filers. In *Robotics Research. The Eleventh International Symposium*, pages 254–266. Springer, 2005.

[TM06] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006.

License