A Practical, Decision-theoretic Approach to Multi-robot Mapping and Exploration

Jonathan Ko, Benjamin Stewart, Dieter Fox, Kurt Konolige[†], and Benson Limketkai[†]

University of Washington, Computer Science & Engineering, Seattle, WA
[†]SRI International, Artificial Intelligence Center, Menlo Park, CA

Abstract

An important assumption underlying virtually all approaches to multi-robot exploration is prior knowledge about their relative locations. This is due to the fact that robots need to merge their maps so as to coordinate their exploration strategies. The key step in map merging is to estimate the relative locations of the individual robots. This paper presents a novel approach to multi-robot map merging under global uncertainty about the robot's relative locations. Our approach uses an adapted version of particle filters to estimate the position of one robot in the other robot's partial map. The risk of false-positive map matches is avoided by verifying match hypotheses using a rendezvous approach. We show how to seamlessly integrate this approach into a decision-theoretic multi-robot coordination strategy. The experiments show that our sample-based technique can reliably find good hypotheses for map matches. Furthermore, we present results obtained with two robots successfully merging their maps using the decision-theoretic rendezvous strategy.

1 Introduction

Efficient exploration of an unknown environment is a fundamental problem in multi-robot coordination. As autonomous exploration and map building becomes increasingly robust on single robots, the next challenge is to extend these techniques to large teams of robots: Our current project aims to field 100 robots. Compared to the problems occurring in single robot exploration, the extension to multiple robots poses several new challenges. These challenges include limited communication between robots, no assumptions about relative start locations of the robots, and dynamic assignment of processing tasks.

The *map merging* problem, *i.e.* the problem of building a map from data collected by different robots, is of utmost importance for efficient multi-robot exploration. This is due to the fact that the coordination of robots requires the availability of a combined world model. Once such a combined model can be generated, Burgard and colleagues [2], for example, show how to coordinate robots by maximizing expected information gain. Zlot *et al*. [14] describe how to deal with limited communication by making the exploration strategies robust w.r.t. communication loss.

The complexity of map merging strongly depends on the assumptions made about the knowledge of the robots' initial locations. Commonly, the initial relative positions of the robots are assumed to be known. Under this assumption, map merging is very similar to map building for single robots, as shown in [10]. Another typical scenario assumes that the robots do not know their relative start locations, but one robot is known to start in the map already built by the other robot. In this case, map merging can be solved mostly by localizing one robot in the other robot's map using a localization approach capable of global localization [10; 5; 12]. In a different setting, Dedoglu and Sukhatme [3] introduced several heuristics using a non-probabilistic, hypothesis rejection approach for topological maps. Their approach assumes that there is an overlap between the maps of the two robots and does not provide a measure of uncertainty. Roy and Dudek [7] propose a rendezvous strategy that aims at guiding robots to the same meeting points so as to determine their relative locations. Their approach is designed for extremely short-range communication and does not coordinate robots.

In this paper we introduce a novel technique that addresses an extremely difficult instance of the multi-robot map merging problem, *i.e.* for completely unknown start locations including a chance that the maps of the two robots do not overlap at all. The approach uses an adapted version of particle filters to estimate the position of one robot in the other robot's partial map. The technique estimates the posterior over robot positions both *inside and outside the partial map*. Thus, it is able to estimate whether or not there is an overlap between the robots' maps. Due to the sequential nature of the estimation process, our approach can be integrated seamlessly into an online, decision-theoretic multi-robot coordination strategy. This strategy trades off the utility of exploration versus the utility of map merging. For large teams of robots, the avoidance of wrong map matches is of utmost importance, since false-positive matches can result in inconsistencies from which a distributed team of robots can not recover. In order to overcome this risk, our robots *actively verify* location hypotheses using a rendezvous strategy (similar in spirit to [7]). Once two robots have verified their relative locations, they merge their maps and coordinate their exploration.

This paper is organized as follows. In the next section, we describe the overall coordination structure for large teams of robots. Section 3 shows how to extend an existing approach to multi-robot coordination to the case of unknown robot locations. The key technical contribution of this paper is given in Section 4, which presents a particle filter based approach to partial map localization. The following section describes experiments supporting the relia-

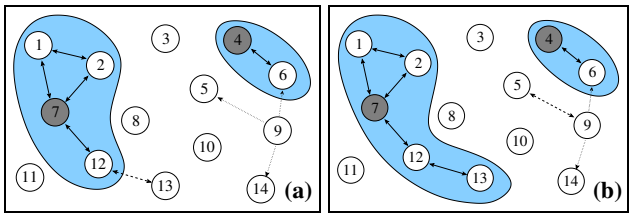

Figure 1: *Dynamic coordination structure at two points in time. Circles indicate robots, solid arrows indicate robots exploring in coordination, dashed lines indicate robots verifying a hypothesis of their relative positions, and dotted lines indicate communication between robots but no valid location hypotheses exist.*

bility of our techniques. Finally, in Section 6, we conclude and discuss directions for future research.

2 Dynamic Coordination Architecture

Our distributed approach to mapping and exploration utilizes a dynamically evolving coordination architecture. At each point in time, the state of the system can be summarized by a graph structure where the nodes are individual robots and edges represent the current interaction between robots (see Fig. 1). Two robots are connected if there is a communication link between them. The key sub-structure of our architecture are *exploration clusters*, indicated by the two shaded areas in Fig. 1. Such clusters contain all robots that can communicate with each other *and* know their relative locations. The robots within each cluster share a common map and coordinate their exploration strategy, using state-of-the-art techniques such as [2; 10]. Each exploration cluster determines one robot responsible for data combination and coordination (robots `4` and `7` in Fig. 2). All map information is frequently spread throughout the cluster. Direct communication between all robot pairs within a cluster is not required for this update.

A crucial situation occurs when a robot moves into the communication range of another robot (dotted arcs in Fig. 1). At this point in time, the robots do not yet know their relative locations. In our approach, the robot within the larger exploration cluster (measured by map size) estimates the position of the other robot within its partial map, using the sensor data collected by the other robot. For example, Fig. 1(a) shows a situation in which robot `6` estimates the location of robot `9` within the map of its cluster. As soon as `9`'s location is uniquely determined, it can be added to the exploration cluster and coordinate with robots `6` and `4`. Unfortunately, as we will describe in Section 4, it is non-trivial to uniquely determine the best map match from the location of a robot in another robot's partial map. On the other hand, keeping track of all possible map matches over time is intractable since the complexity is exponential in the number of robots. To overcome the complexity problem and the danger of false-positive matches, we propose a very simple, effective method: Robots must *actively verify* their relative locations before merging maps. To do this one robot first generates a hypothesis of another robot's location. The correctness of this hypothesis is resolved by the two robots arranging to meet one another at a rendezvous point. For example, the dashed arc in Fig. 1(a) indicates that robot `12` and `13` are currently verifying a location hypothesis. If the robots fail to meet, the hypothesis is rejected and they continue with the hypothesis generation phase. Otherwise, robot `13` can be added to the exploration cluster, as shown in Fig. 1(b). Whenever robots lose contact, they explore independently until they get back in communication range.

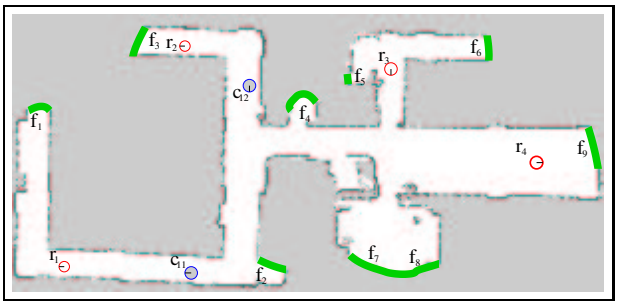

Figure 2: *Coordination example: Shown is a partial map built by an exploration cluster of four robots ($r_1, \ldots, r_4$). Additionally, two location hypotheses (c_{11}, c_{12}) have been generated for robot c_1. The map has nine exploration frontiers ($f_1, \ldots, f_9$).*

3 Decision-theoretic Coordination

This section describes the decision-theoretic framework used to coordinate robots within an exploration cluster. We assume that the robots within the cluster share a map and the positions r_i of all robots in the partial map are known. Fig. 2 shows an exploration cluster of four robots sharing a partial occupancy grid map. Areas in which unoccupied, explored grid cells are next to unexplored areas serve as exploration frontiers f_i [13; 2]. The number of frontiers is reduced by merging neighboring frontier grid cells, as shown in the figure. If there are other robots in the communication range of the cluster, then c_{ij} denotes the j-th hypothesis for robot c_i's position within the partial map of the cluster. At any point in time, each robot in the exploration cluster can be assigned either to a frontier or to a hypothesized location of another robot. Coordination can be phrased as the problem of finding the assignment that maximizes a utility-cost trade-off, similar to the approach used in [2], with the extension of hypothesis verification. More specifically, let θ denote an assignment that determines which robot should move to which target (frontiers and hypotheses). Each robot is assigned to exactly one target and $\theta(i,j) = 1$ if the i-th robot in the exploration cluster is assigned to the j-th target. Among all assignments we choose the one that maximizes expected utility minus expected cost:

$$\theta^* \quad = \quad \underset{\theta}{\operatorname{argmax}} \sum_{(i,j)\in\theta} \theta(i,j)\,(\mathrm{U}(i,j) - \mathrm{C}(i,j)) \quad (1)$$

The cost and utility of each robot target pair (i,j) can be computed as follows.

Cost: If the target is a frontier then the cost is given by the minimum cost path from the robot's position r_i to the

frontier position f_j. Minimal cost paths can be computed efficiently by A^* search. For hypothesis verification, the cost is given by the minimal path to a meeting point between the robots plus the cost to establish whether the two robots actually meet or not.

$$\mathrm{C}(i,j) \;=\; \begin{cases} \mathrm{dist}(r_i, f_j) & \text{if target is frontier } f_j \\ \mathrm{verify}(r_i, c_{kj}) & \text{if target is hypothesis } c_{kj} \end{cases} \quad (2)$$

Utilities: For simplicity, we assume that all robots have the same exploration capabilities, *i.e.* the utility only depends on the type of target and not the robot. If the target is a frontier, then the utility can be estimated by the expected area the robot will explore at that frontier [2]. If the target is a location hypothesis, say c_{kj}, then the utility is given by the expected utility of meeting robot c_j. The function *coord* estimates this utility by measuring the map size of the other robot plus the expected utility of coordinated exploration versus independent exploration. Since it is not known whether the other robot is at the location hypothesis, the utility of meeting is weighted by the probability of the hypothesis, denoted $p(c_{kj})$.

$$\mathrm{U}(i,j) \;=\; \begin{cases} \mathrm{explore}(r_i, f_j) & \text{if target is frontier } f_j \\ p(c_{jk})\mathrm{coord}(c_j) & \text{if target is hypothesis of robot } c_j \end{cases} \quad (3)$$

In most cases, all robots are assigned to frontiers, thereby extending the explored area. However, if another robot enters the communication range of the cluster, the relative position of this robot is estimated and subsequently verified by meeting with the robot. In the next section, we will describe how to efficiently estimate the position of a robot in a partial map.

4 Partial Map Localization

This section describes how to sequentially determine location hypotheses by localizing one robot in another robot's partial map. Since this problem is very similar to the regular localization problem, we will first describe the recursive Bayes filter, which underlies virtually all probabilistic robot localization techniques [1; 6]:

$$p(x_t \mid z_{1:t}, u_{1:t-1}) \;=\; \alpha_t\, p(z_t \mid x_t) \cdot \int p(x_t \mid x_{t-1}, u_{t-1})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-2})\, dx_{t-1}, \quad (4)$$

Here $z_{1:t}$ is the history of all sensor measurements obtained up to time t, and $u_{1:t-1}$ is the control information. Furthermore, α_t is a normalizing constant which ensures that the belief over the entire state space sums up to one. In the context of robot localization, the term $p(x_t \mid x_{t-1}, u_{t-1})$ is a probabilistic model of robot motion, and $p(z_t \mid x_t)$ describes the likelihood of making observation z_t given the robot's location x_t and a map of the environment (see [6] for details). In our implementation, maps are occupancy grid maps and observations are laser range-scans.

A straightforward approach to partial map localization would use the recursive Bayes filter (4) to estimate a robot's position both inside and outside the partial map. The likelihood of observations outside the map can be approximated by a fixed value $p(z_t \mid \text{outside})$, which can be learned from previous data. Unfortunately, since only a small part of the environment might be represented in the partial map, such an approach would require estimating a robot's location in a potentially huge state space. Our solution to this problem is based on the fact that we are only interested in those robot trajectories that have an overlap between the two maps, *i.e.* trajectories for which the other robot is in the partial map at some point in time.

More specifically, non-overlapping trajectories are not represented explicitly but summarized by a single state n_t that measures the probability that the robot was not in the partial map so far. To do so, we adapt the recursive update rule (4) so as to only track positions x_t that correspond to overlapping trajectories.

$$p(x_t \mid z_{1:t}, u_{1:t-1}) \;=\; \alpha_t\, p(z_t \mid x_t) \cdot \Big[\int p(x_t \mid x_{t-1}, u_{t-1})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-2})\, dx_{t-1} + p(x_t \mid n_{t-1}, u_{t-1})\, p(n_{t-1} \mid z_{1:t-1}, u_{1:t-2}) \Big] \quad (5)$$

Note that some of the positions x_t might be outside the partial map, since the only constraint is that their trajectory up to time t had an overlap with the map. As suggested above, we use a fixed value $p(z_t \mid \text{outside})$ to determine the likelihood of z_t for such positions outside the map. The additional term on the right side of (5) models the fact that the robot can enter the map at any iteration for the first time. More specifically, if the robot has not been in the partial map so far, it enters the map in the current iteration with a fixed probability ε:

$$p(x_t \mid n_{t-1}, u_{t-1}) \;=\; \varepsilon\; p(x_t \mid f_{t-1}, u_{t-1}) \quad (6)$$

Here we assume that the positions x_t immediately after entering the map only depend on the positions of the frontier cells f_{t-1} of the partial map, that is we use the transitions from unexplored to free space as entry points into the partial map (see also Fig. 3(b)–(d)). At each iteration, the probability of non-overlapping trajectories is updated based on the observation likelihood and the fact that a fraction ε of trajectories moved into the map:

$$p(n_t \mid z_{1:t}, u_{1:t-1}) \;=\; \alpha_t\, p(z_t \mid \text{outside})(1-\varepsilon)\, p(n_{t-1} \mid z_{1:t-1}, u_{1:t-2}) \quad (7)$$

Again, $p(z_t \mid \text{outside})$ denotes the likelihood of observing z_t outside the partial map. The normalization factor α_t is computed so that the probabilities over n_t and all locations x_t sum up to one. The effect of this normalization is such that if the observation is very likely inside the map, then the outside likelihood $p(z_t \mid \text{outside})$ is lower and the probability of n_t decreases while the probability for overlapping trajectories x_t increases.

Implementation as particle filter

We will now describe how to implement the recursive Bayes filter for partial map localization using particle filters (see also [6; 4]). In a nutshell, particle filters represent posteriors over a robot's position by sets $S_t = \{\langle x_t^{(i)}, w_t^{(i)} \rangle \mid$

1.	**Inputs:** $S_{t-1} = \{\langle x_{t-1}^{(i)}, w_{t-1}^{(i)}\rangle \mid i = 1, \ldots, N\}$, n_{t-1}, control information u_{t-1}, observation z_t, partial map $\mathcal{M}_t$, probability of entering partial map ε, number of entry point samples N_ε	
2.	$S_t := \emptyset$	***// Initialize***
3.	**for** all samples $\langle x_{t-1}^{(i)}, w_{t-1}^{(i)}\rangle$ in S_{t-1} **do**	***// Prediction step***
4.	sample $x_t^{(j)}$ from $p(x_t \mid x_{t-1}^{(i)}, u_{t-1})$	*// Predict next position using motion*
5.	$S_t := S_t \cup \{\langle x_t^{(j)}, w_{t-1}^{(i)}\rangle\}$	*// Insert into next set*
6.	**for** $i := 1, \ldots, N_\varepsilon$ **do**	***// Generate N_ε samples at entry points***
7.	sample $x_t^{(i)}$ from an entry point into the map	*// Entry points are given by transition from free space to unexplored*
8.	$w_t^{(i)} = \frac{\varepsilon\, n_{t-1}}{N_\varepsilon}$, $S_t := S_t \cup \{\langle x_t^{(i)}, w_t^{(i)}\rangle\}$	*// Adjust weights accordingly and insert into set*
9.	$n_t = (1 - \varepsilon)\, n_{t-1}$	*// Subtract fraction ε migrated from non-overlapping trajectories into map*
10.	**for** all samples $\langle x_t^{(i)}, w_t^{(i)}\rangle$ in S_t **do**	***// Integrate observation into individual samples***
11.	**if** $x_t^{(i)} \in \mathcal{M}_t$ **then** $w_t^{(i)} := w_{t-1}^{(i)}\, p(z_t \mid x_t^{(i)})$ **else** $w_t^{(i)} := w_{t-1}^{(i)}\, p(z_t \mid \text{outside})$	
12.	$n_t = n_t\, p(z_t \mid \text{outside})$	*// Integrate observation into non-overlapping trajectories*
13.	$\alpha_t^{-1} = n_t + \sum_{i=1,\ldots,N+N_\varepsilon} w_t^{(i)}$	*// Compute normalization factor*
14.	resample samples in S_t using $p(z_t \mid x_t^{(i)})$ only	*// Includes readjusting the weights according to partial resampling*
15.	determine location hypotheses	*// Hypotheses are determined using a grid over the state space*

Table 1: Outline of particle filter based implementation of partial map localization.

$i = 1, \ldots, N\}$ of N weighted samples distributed according to the posterior. Here each $x_t^{(i)}$ is a sample (or state), and the $w_t^{(i)}$ are non-negative numerical factors called *importance weights*, which sum up to one. Sets at time t are generated from previous sets S_{t-1} by a sampling procedure often refereed to as SISR, sequential importance sampling with re-sampling. SISR implements the recursive Bayes filter update rule in a three stage process: First, draw samples $s_t^{(i)}$ from the previous sample set using the importance weights $w_t^{(i)}$, then draw for each such sample a new state from the predictive distribution $p(x_t \mid x_{t-1}^{(i)}, u_{t-1})$, and finally weight these new states/samples proportional to the observation likelihood $p(z_t \mid x_t)$. The last step, importance sampling, adjusts for the fact that samples are not drawn from the actual posterior distribution but from the predictive distribution.

Particle filters can be used to implement the partial map algorithm as follows (Table 1). In the first iteration, at $t = 0$, when there is no knowledge about the relative locations of the two robots, the samples are spread uniformly throughout the partial map. These samples initialize overlapping trajectories x_0, and they are weighted so that they sum up to $(1 - n_0)$. n_0, the probability that the other robot initially is not in the partial map, can be set according to an estimate of the ratio between the sizes of the partial map and the entire environment. At each iteration, the algorithm updates the sample set S_t and the probability of non-overlapping trajectories n_t based on the most recent observation z_t and motion information u_{t-1} (see Tab. 1). The algorithm additionally takes $\mathcal{M}_t$, the current partial map, as input. (5) is implemented in steps 3–8 of the algorithm. In steps 3–5, new samples are generated based on the previous sample set. Samples entering the map for the first time are added in steps 6–8. The weights of these samples follow directly from (6). Line 9 adjusts the probability of non-overlapping trajectories according to (7). Steps 10–12 integrate the observation z_t into all relevant quantities. The normalization factor is determined in the following step. The resampling step, a crucial part of each particle filter, will be described in the next section. Finally, the algorithm determines hypotheses for the robot's location by clustering the samples. In our current implementation, this is done by overlaying a grid over the state space, searching for the most likely grid cell (as determined by the samples within each cell) and then performing local averaging in this cell.

After each iteration, the weighted samples accurately represent the posterior of the other robot's position given that there is an overlap between the maps of the two robots. The probability of no such overlap is given by n_t.

Partial resampling

In the basic particle filter, all samples are frequently resampled based on their accumulated weights [4]. Unfortunately, in our context, such a resampling does not work since the weights of the samples may differ by orders of magnitude. For example, the probability that the robot enters the partial map at any specific point in time is very small (ε is set to 0.01 in our experiments), resulting in extremely small weights for the entry point samples (step 8 in the algorithm). Hence, these samples are very likely not to be chosen during the next resampling step, thereby making it almost impossible to detect situations in which the robot enters the map for the first time. Fortunately, it is not necessary to resample according to the current weights of the samples, as shown by [11]. In the context of importance sampling, if we have samples with weights $w^{(i)}$ and resample them w.r.t. to a weighting function $f^{(i)}$, then the weights of the samples have to be adjusted to values proportional to $w^{(i)}/f^{(i)}$. This is consistent with the normal resampling case, where we resample w.r.t. to the weights $w^{(i)}$ and get samples with uniform weights. If we resample

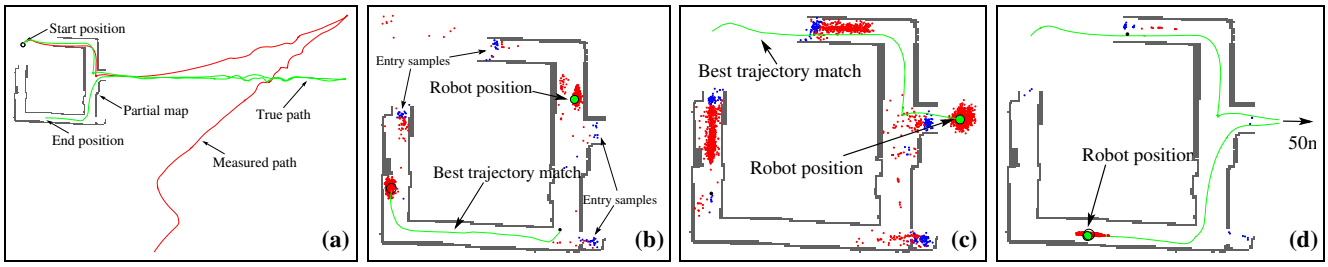

Figure 3: *Partial map localization: (a) Partial map and trajectory of the other robot. (b)–(d) Sample sets at different points in time. Shown are the sets after partial resampling,* i.e. *the samples have different weights. The pictures also show the entry point samples and the most likely hypothesis for the other robot's position, along with its path attached to this hypothesis. (a) After only short overlap, the best match is not yet correct. The summed probability of all samples inside the map is* 0.497*. (b) The robot exits the map, but already determined the correct match. Now, the probability of being inside the map dropped to* 0.00037*, since all high-weight samples just exited the map. (c) After moving about 50m outside the map, the robot returned and the match is correct (probability of this match is* 0.799*).*

Figure 4: *Sample sets after resampling using the complete weights of the samples shown in Fig. 3 (b)–(d). It becomes clear that the approach puts high weights on the correct hypotheses.*

samples inside the map w.r.t. to the observation likelihood $p(z_t \mid x_t^{(i)})$, then we only have to divide the weights of the resulting samples by this likelihood. Note that even though this step reverses the likelihood multiplication in step 11 of the algorithm, it is still necessary to compute the weights in step 11. This weight update is needed to weight samples inside the partial map against samples outside the map (which are weighted by $p(z_t \mid \text{outside})$). As a result of this resampling procedure, all samples do *not* have the same weights, but carry the "non-resampled" weights with them into the next iteration. Hence, even though our approach does not resample the complete sample set at each iteration, it still keeps track of the correct posterior density. Furthermore, it is possible to generate the fully resampled set at any point in time (cf. 3 and 4).

Intuitively, the key advantage of this partial resampling step is that samples that enter the map with a small weight are not immediately "deleted" in the next resampling step, but they are given a chance to accumulate higher weights if they predict observations well inside the map. Another very important advantage of this approach is that samples outside the partial map are not deleted but tracked until they re-enter the map.

5 Experiments

In these experiments we test different aspects of our approach to map merging and its integration into the decision-theoretic coordination.

5.1 Map merging

Fig. 3 illustrates a typical partial map localization run using a map built from 30m of robot motion. Fig. 3(a) shows the partial map with the complete path of the other robot (as measured by the robot's wheel encoders and as determined by localization in the complete map). It is clear that occupancy map matching as described in [8] is very likely to fail in finding good matches in such environments, since the trajectory of the other robot is highly corrupted by noise. In the beginning of this example, all samples are spread uniformly inside the partial map. The start position of the other robot is outside the map. Fig. 3(b)–(d) show the partially resampled sample sets at different points in time. The corresponding sets after full resampling are shown in Fig. 4. The sequence shows that our approach is able to find the correct match even when the other robot starts outside the map, enters, exits, and then re-enters the partial map.

To evaluate our approach more systematically, we built 15 partial maps based on data collected in three indoor environments. We then collected additional data in the same environments and randomly chose sub-trajectories from this data. Each partial map was paired with 10 sub-trajectories resulting in 150 map-trajectory pairs (the maps built by robot `A` and the trajectories generated by robot `B`). For each pair, we determined the quality of our partial map localization approach using a precision-recall measure adapted to the sequential nature of the decision making problem. To see, consider that at each iteration of the particle filter, robot `A` determines the probability of the most likely hypothesis for `B`'s position in its map. In our evaluation, `A` considers a hypothesis to be valid if its probability exceeds a specific threshold θ. *Precision* measures which fraction of these valid hypotheses are correct. Correctness is tested by comparing the position of the hypothesis to a reference path computed offline in the complete map of the environment. To determine recall, we checked at what times robot `B` was in robot `A`'s partial map. *Recall* measures the fraction of this time for which robot `A` generated a correct hypothesis, *i.e.* at the correct position and with probability above the threshold θ.

We used the 150 map-trajectory pairs to test the influence of partial resampling on the performance of map merging. This was done by evaluating the precision recall trade-off for different values of $p(z \mid \text{outside})$, the likelihood of sensor measurements outside the partial map. This likelihood

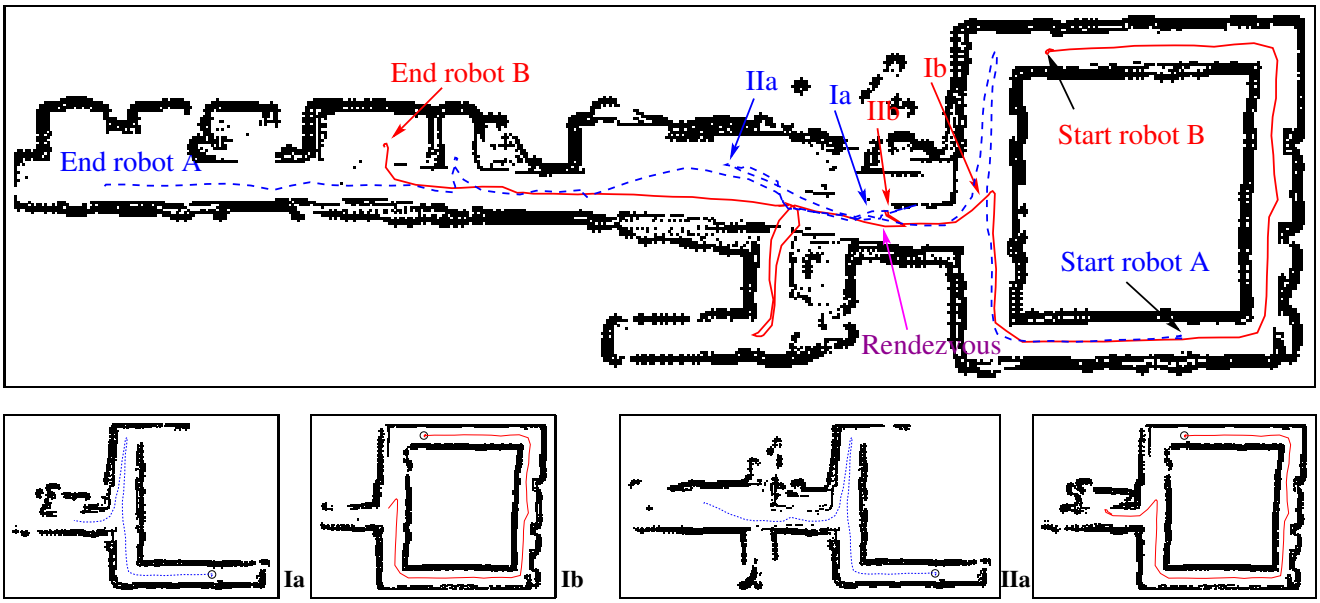

Figure 6: *Coordinated exploration: Robots* `A` *and* `B` *start at unknown locations and explore independently. The trajectories of* `A` *and* `B` *are shown as dotted and solid lines, respectively. After some time, the robots reach positions* `Ia` *and* `Ib`*, and* `A` *estimates* `B`*'s location in its map. The corresponding maps are shown in Ia and Ib. The overlap between the two maps is not sufficient to create a hypothesis with probability above* $\theta = 0.9$*. Both robots keep exploring until, at positions* `IIa` *and* `IIb`*,* `A` *finds a very likely hypothesis for* `B`*'s position. Both robots move to the meet point and verify the hypothesis. The maps are merged and the robots start coordinated exploration.* `A` *moves to the left and* `B` *first moves into the small hallway in the lower part.*

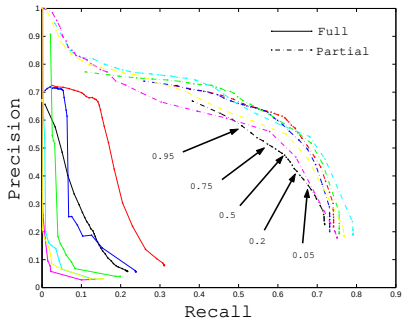

Figure 5: *Precision* vs. *recall: Each point represents an average over 150 pairs of partial maps and trajectories. Each curve shows the trade-off for different thresholds* θ *(0.0001-0.999). The dashed lines indicate results obtained with partial resampling for different values of* $p(z \mid \text{outside})$ *(the highest value is nine times greater than the lowest value). Solid lines represent the corresponding results obtained with the original resampling approach.*

is a crucial value for partial localization since it determines the ratio between weights of samples inside the partial map and those outside the map. The solid and dashed curves in Fig. 5 show the results obtained for six different likelihoods using partial resampling and the original resampling, respectively. Each curve gives the trade-off for different thresholds θ (0.0001–0.999). Note that values to the upper right represent better trade-offs. The curves show that partial resampling is clearly superior to full resampling. Partial resampling demonstrates better precision/recall rates and robustness against parameter changes. The curves also show that for the best $p(z \mid \text{outside})$, our approach achieves a precision of 75% and at the same time a recall of 45%, *i.e.* 75% of all hypotheses exceeding the threshold are correct, and during 45% of the time robot `B` was in the map, the probability of the correct location was above the threshold and had the highest likelihood among all locations [1]. For the same likelihood, our approach can achieve a recall rate of 70%, but at the cost of lowering the precision to 55%. These results are extremely encouraging since they were obtained with rather small maps obtained from 20–30m of robot motion. Obviously, for larger maps, the precision and recall can both be increased. Furthermore, our approach to hypothesis verification additionally increases the robustness against false matches.

5.2 Integration into coordination

In this preliminary experiment we demonstrate the integration of partial map merging into the multi-robot coordination structure. To do so, two Pioneer DX2 robots were placed at different, unknown locations in our environment (see Fig. 6). The task of the robots was to explore the environment, merge their maps when possible, and then perform coordinated exploration. The picture shows that the two robots successfully completed the individual steps of this task. Note that not all steps were fully automated. Robot detection was determined manually by checking whether the two robots moved within less than one meter. We are currently implementing unique robot

[1] Note that recall can not be expected to be much higher, since the robot has to move for several meters inside the map before a match can be found

detection using signatures in laser range-scans and a coordinated motion pattern between two robots.

6 Conclusions and Future Research

We have presented a novel approach to coordinated multi-robot exploration under global uncertainty about the robot's relative start locations. The key technical innovation of this approach is an adapted particle filter that allows pairs of robots to efficiently and sequentially estimate their relative locations during the exploration process. The approach estimates the posterior over robot positions both inside and outside the partial map. Thus, a robot does not only estimate another robot's position within its partial map, but it can also estimate the probability of an overlap between the maps. In contrast to an alternative approach using map matching, our technique can estimate map matches sequentially, *i.e.* the estimates can be updated whenever new data arrives. The experimental results show that this approach works very well even on small map patches based on 20m of robot motion.

We also show how to seamlessly integrate this approach to map merging into a decision-theoretic multi-robot coordination strategy. In order to overcome the risk of false-positive map matches, the robots actively verify location hypotheses using a rendezvous strategy. Initial experiments using two robots show that our system is able to explore environments even when the start locations of the robots are unknown. Despite these encouraging results, this is only a first step toward the robust deployment of large robot teams. However, we strongly believe that the framework described in this paper scales toward the scenario of distributed exploration of very large indoor environments. We are currently working on a complete implementation and thorough experimental evaluation of our architecture.

So far, we assume that robots within an exploration cluster can always communicate with each other. The consideration of communication for the exploration strategy is an important aspect of future research. Our approach to partial map localization can sequentially estimate another robot's location. However, as described in this paper, the technique assumes that the partial map does not change during the localization process. Obviously, this assumption is not valid. We are currently investigating how to adjust the approach to dynamically changing maps. First tests indicate that good results can be achieved by simply growing the map during the partial localization process (deleting samples that are in occupied areas as the map grows).

Another key problem in partial map merging has not been addressed in this paper. This is the question of how to determine the likelihood of observations *outside* the partial map. Currently, we set a fixed value for this likelihood. The experiments show that our approach can achieve good results even for highly varying settings of this likelihood. In [9] we present an approach that uses a hierarchical Bayesian model to estimate the structure of an environment. This structure is used to better estimate the likelihoods of sensor measurements outside of partial maps. Experiments using a feature based sensor model show that this approach can increase the quality of the estimation process.

Acknowledgments

This work has partly been supported by the NSF under grant number IIS-0093406 and by DARPA's SDR Programme (contract number NBCHC020073).

References

[1] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. John Wiley, 2001.

[2] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2000.

[3] G. Dedeoglu and G.S. Sukhatme. Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Proc. of the 5th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2000.

[4] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo in Practice*. Springer-Verlag, New York, 2001.

[5] J.W. Fenwick, P.M. Newman, and J.J. Leonard. Cooperative concurrent mapping and localization. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.

[6] D. Fox. Adapting the sample size in particle filters through KLD-sampling. *International Journal of Robotics Research (IJRR)*, 22, 2003.

[7] N. Roy and G. Dudek. Collaborative exploration and rendezvous: Algorithms, performance bounds and observations. *Autonomous Robots*, 11(2), 2001.

[8] A.C. Schultz and A. William. Continuous localization using evidence grids. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 1998.

[9] B. Stewart, J. Ko, D. Fox, and K. Konolige. The revisiting problem in mobile robot map building: A hierarchical Bayesian approach. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2003.

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

[11] S. Thrun, J. Langford, and V. Verma. Risk sensitive particle filters. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2001.

[12] S.B. Williams, G. Dissanayake, and H. Durrant-Whyte. Towards multi-vehicle simultaneous localisation and mapping. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.

[13] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proc. of the Second International Conference on Autonomous Agents*, 1998.

[14] R. Zlot, A. Stentz, M. Bernardine Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.