INVITED PAPER

Distributed Multirobot Exploration and Mapping

Robots exploring an indoor environment can exchange position and orientation data to locate themselves in shared maps, then meet at common locations to verify that the maps are correct.

By Dieter Fox, Jonathan Ko, Kurt Konolige, Benson Limketkai, Dirk Schulz, and Benjamin Stewart

ABSTRACT | Efficient exploration of unknown environments is a fundamental problem in mobile robotics. We present an approach to distributed multirobot mapping and exploration. Our system enables teams of robots to efficiently explore environments from different, unknown locations. In order to ensure consistency when combining their data into shared maps, the robots actively seek to verify their relative locations. Using shared maps, they coordinate their exploration strategies to maximize the efficiency of exploration. This system was evaluated under extremely realistic real-world conditions. An outside evaluation team found the system to be highly efficient and robust. The maps generated by our approach are consistently more accurate than those generated by manually measuring the locations and extensions of rooms and objects.

KEYWORDS | Exploration; mapping; multirobot; simultaneous localization and mapping (SLAM)

I. INTRODUCTION

Efficient exploration of unknown environments is a fundamental problem in mobile robotics. As autonomous exploration and map building becomes increasingly robust on single robots, the next challenge is to extend these techniques to teams of robots. Compared to the problems occurring in single robot exploration, the extension to multiple robots poses several new challenges, including: 1) coordination of robots; 2) integration of information collected by different robots into a consistent map; and 3) dealing with limited communication.

Coordination: Increasing efficiency is one of the key reasons for deploying teams of robots instead of single robots. The more robots that explore an environment, the more important the coordination between their actions becomes. The difficulty of the coordination task strongly depends on the knowledge of the robots. If the robots know their relative locations and share a map of the area they explored so far, then effective coordination can be achieved by guiding the robots into different, nonoverlapping areas of the environment [1], [2], [29], [37]. This can be done by assigning the robots to different exploration frontiers, which are transitions from explored free space to unexplored areas [1], [36]. However, if the robots do not know their relative locations, then it is far less obvious how to effectively coordinate them, since the robots do not share a common map or frame of reference.

Map merging: In order to build a consistent model of an environment, the data collected by the different robots has to be integrated into a single map. Furthermore, such an integration should be done as early as possible, since the availability of a shared map greatly facilitates the coordination between robots. If the initial locations of the robots are known, map merging is a rather straightforward extension of a single robot mapping [6], [20], [32], [35]. This is due to the fact that the data traces of the individual robots can be treated as if they were collected by a single robot. Consistent integration of the data when the robots do not know their relative locations is more difficult, since it is not clear how and where the robots' traces should be connected.

Limited communication: During exploration of large-scale environments, communication between the robots

Manuscript received June 1, 2005; revised June 1, 2006. This work was supported in part by the National Science Foundation under CAREER Grant IIS-0093406 and in part by the Defense Advanced Research Project Agency under Contract NBCHC020073.

D. Fox, **J. Ko**, **B. Limketkai**, and **B. Stewart** are with the Computer Science and Engineering Department, University of Washington, Seattle, WA 98195-2350 USA (e-mail: fox@cs.washington.edu; jonko@cs.washington.edu; bensonl@cs.washington.edu; ben.d.stewart@gmail.com).
K. Konolige is with the Artificial Intelligence Center, SRI International, Menlo Park, CA 94025 USA (e-mail: konolige@ai.sri.com).
D. Schulz is with the Department of Computer Science III, University of Bonn, Bonn, Germany (e-mail: schulz@iai.uni-bonn.de).

Digital Object Identifier: 10.1109/JPROC.2006.876927

and a control station might fail. To achieve robustness against such failures, each robot has to be able to explore on its own, i.e., without guidance by a central control node. Furthermore, groups of robots should be able to coordinate their actions without the need of a central control node, and each robot should be able to take over the task of coordination.

In this paper, we present an integrated multirobot mapping and exploration system that addresses all these challenges. The approach enables teams of robots to efficiently build highly accurate maps of unknown environments, even when the initial locations of the robots are unknown. In order to avoid wrong decisions when combining their data into shared maps, the robots actively verify their relative locations. Using shared maps, they coordinate their exploration strategies to maximize the efficiency of exploration. Our system was evaluated thoroughly by an outside evaluation team. The results of this test showed that our approach is highly efficient and robust. The maps generated by our robots are consistently more accurate than those generated by manually measuring the locations and extensions of rooms and objects.

This paper is organized as follows. In the next section, we provide an overview of our multirobot coordination technique, followed by a description of an approach to estimating relative locations between robots. Then, in Section IV, we show how the data collected by multiple robots can be integrated into a consistent map of an environment. The following section describes experiments supporting the reliability of our techniques. We conclude in Section VI.

II. DECISION-THEORETIC COORDINATION ARCHITECTURE

We will now discuss the concept underlying our multirobot coordination technique; implementation details will be provided in Section V.

A. Related Work

Virtually all existing approaches to coordinated multirobot exploration assume that all robots know their locations in a shared (partial) map of the environment. Using such a map, effective coordination can be achieved by extracting exploration frontiers from the partial map and assigning robots to frontiers based on a global measure of quality [1], [2], [29], [31], [37]. As illustrated in Fig. 1, exploration frontiers are borders of the partial map at which explored free space is next to unexplored areas [1], [24], [36]. These borders thus represent locations that are reachable from within the partial map *and* provide opportunities for exploring unknown terrain, thereby allowing the robots to greedily maximize information gain [17], [34]. To measure the quality of an assignment of robots to frontiers, the overall travel distance combined

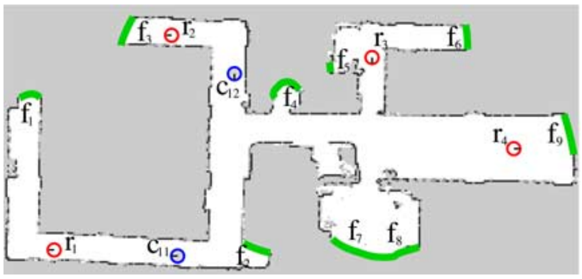


Fig. 1. ***Coordination example: Partial map built by exploration cluster of four robots (red circle $r_1, \ldots, r_4$). Additionally, two location hypotheses (blue circles c_{11}, c_{12}) have been generated for robot c_1, which is not yet part of cluster. Map has nine exploration frontiers ($f_1, \ldots, f_9$), indicated by green lines.***

with an estimate of the unexplored area at each frontier proved to be highly successful in practice [1], [29].

The assumption of the availability of a shared map, however, severely restricts the scenarios that can be handled by such an exploration strategy. For instance, a unique globally consistent map can be generated only if the robots know their relative locations. If the robots do not know their relative locations, then it is not clear how they can combine their maps into a global, shared map. Knowledge about relative locations is readily available only if all robots start at the same location or have sensors that provide location estimates in a global frame of reference. While the latter case can hold when using a global positioning sensor (GPS) for outdoor exploration [26], there exists no GPS for indoor environments. Thus, in order to deal with more general exploration settings, the robots must be able to handle uncertainty in their relative locations, which directly translates into uncertainty in how to combine their maps.

In a full Bayesian treatment, the robots could estimate posterior probability distributions over their relative locations and then coordinate their actions based on the resulting distribution over shared maps. While such an approach could lead to a highly effective exploration strategy, it does not scale well, since the number of possible relative locations, and thus maps, grows exponentially in the number of robots. To avoid this complexity, virtually all approaches to multirobot mapping under position uncertainty let the robots explore independently until they have reliable estimates of their relative locations; at which time, their maps are merged and the robots start to coordinate their exploration strategies [3], [6], [14], [16], [32], [35]. To estimate relative locations, Howard and colleagues rely on the robots' ability to detect each other [14]. Here, all robots explore independently of each other until one coincidentally detects another robot. The robots use such detections to determine their relative location, based on which they combine their maps. While such an approach scales well in the number of robots, it can result in inefficient exploration, since it can take arbitrarily long until robots coincidentally detect each other. For instance, if one robot follows the path of the other robot without knowing, both robots might explore the complete map without ever detecting each other. Other approaches establish relative locations between pairs of robots by estimating one robot's location in another robot's map. This is typically done under the assumption that one robot starts in the map already built by the other robot [6], [32], [35] or that there exists an overlap between the partial maps [3]. Since these techniques do not verify location estimates, they might erroneously merge maps, which typically results in inconsistent maps.

Our approach combines and extends these ideas in order to generate an efficient and robust exploration system. In contrast to [3], [6], [32], and [35], our techniques makes *no assumptions about the relative locations of robots*. Furthermore, it adds robustness by verifying hypotheses for the relative location of robots. Similar to [14], this is done by using robot detections. However, in contrast to [14], these detections are not coincidental; they are pursued actively.

B. Decision-Theoretic Coordination

Our technique for exploration with unknown start locations integrates robot detections into a Bayesian decision-theoretic exploration strategy. In a nutshell, our system works as follows. Initially, the robots might not know their relative locations. In such a case, each robot explores on its own, mapping an increasingly large portion of the environment. As soon as two robots can communicate, they start to exchange sensor data and estimate their relative location. Once they have a good hypothesis for their relative location, they actively verify this hypothesis using a rendezvous technique. If successful, the robots form an exploration cluster; they combine their data into a shared map and start to coordinate their exploration actions. On the other hand, if the relative location hypothesis turns out to be wrong, the robots continue to explore independently and exchange sensor data to refine their estimates of their relative location. During exploration, the size of exploration clusters increases as more robots determine their relative locations, ending in a single cluster containing all robots.

As long as a robot is not part of an exploration cluster, it individually explores an environment by moving to the closest exploration frontier in its partial map [17], [36]. To coordinate the robots within an exploration cluster, we extend the decision-theoretic approaches of [1], [2], [29], and [37] to the case of relative position uncertainty [16]. To do so, we assume that the robots within an exploration cluster share a map and that the positions r_i of all robots in the shared map are known. Fig. 1 shows an exploration cluster of four robots sharing a partial occupancy grid map. Exploration frontiers f_i are indicated by thick green lines. The figure also shows hypotheses c_{11} and c_{21} for the location of a robot not yet part of the cluster. In general, let c_{ij} denote the ith hypothesis for the unknown location of robot j. $p(c_{ij})$ is the probability that robot j actually is at this hypothesis (how hypotheses and their probabilities are determined will be described in Section III).

The robots in an exploration cluster trade off exploring unknown terrain and verifying hypotheses for the locations of other robots. Hypothesis verification is done by sending one of the robots to the hypothesis and physically testing whether there actually is another robot. In our system, similar to [14], robot detections are performed by marking robots with highly reflective tape and using laser range finders to detect these markers. Once a location hypothesis is verified, the data of this robot can be added to the cluster map and the robot can participate in coordinated exploration. At any point in time, each robot in the exploration cluster can be assigned either to an exploration frontier or to a hypothesized location of a robot outside the cluster. Coordination between the robots can be phrased as the problem of finding the assignment from robots to frontiers and hypotheses that maximizes a utility–cost tradeoff. To see, 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 ith robot in the exploration cluster is assigned to the jth target. Among all assignments we choose the one that maximizes expected utility minus expected cost

$$\theta^* = \operatorname*{argmax}_{\theta} \sum_{(i,j)\in\theta} \theta(i,j)\left(U(i,j) - C(i,j)\right). \tag{1}$$

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_k. Minimal cost paths can be computed efficiently by an 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

$$C(i,j) = \begin{cases} \text{dist}(r_i, f_k), & \text{if } j\text{th target is frontier } f_k \\ \text{verify}(r_i, c_{pq}), & \text{if } j\text{th target is hypothesis } c_{pq}. \end{cases} \tag{2}$$

Utilities: If the target is a frontier, then the utility is given by the expected area the robot will explore at that frontier. This area is estimated by the size of the unknown area visible from the frontier [1]. If the target is a location hypothesis, say c_{pq}, then the utility is given by the expected utility of meeting robot r_q. 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 by $p(c_{pq})$

$$U(i,j) = \begin{cases} \text{explore}(r_i, f_k), & \text{if } j\text{th target is frontier } f_k \\ p(c_{pq})\text{coord}(r_q), & \text{if } j\text{th target is hypothesis } c_{pq} \end{cases}. \tag{3}$$

Once the pairwise utilities and costs are computed, we use a linear program solver to find the optimal assignment. Finding optimal assignments can be performed in time $O(mn)$, where m is the number of robots and n is the number of goals [11]. In exploration scenarios involving up to six robots, we found the overall computation time for this decision step to be negligible compared to the other cost involved in exploration (less than 1 s). Using the tradeoff between (2) and (3), robots typically move to exploration frontiers and only choose a hypothesis as a target if it is not too far away and its probability is very high.

III. ESTIMATING RELATIVE POSITIONS

We now discuss an algorithm for sequentially estimating the relative locations between pairs of robots exploring an environment [16]. In order to perform this estimation, robots exchange laser range scans and odometry motion information whenever they are in communication range. Our approach considers only pairs of robots since the complexity of estimating relative locations is exponential in the number of robots considered jointly. One approach to estimating the overlap between partial maps of two robots is to compute the correlations between the maps for each possible overlap. Unfortunately, such an approach does not adequately handle uncertainty in mapping and does not lend itself to an incremental implementation. We overcome this problem by using an adapted particle filter in combination with a predictive model of indoor environments in order to sequentially determine whether and how the partial maps of two robots overlap.

A. Particle Filter for Partial Map Localization

Existing approaches to robot localization have only addressed the problem of localizing a robot in a complete map of an environment. Particle filters have been applied with great success to this problem [7], [8], [15], [21]. The main difference between localizing a robot in a complete and in a partial map of an environment is due to the fact that the robot might not be inside the partial map and that the robot can enter or exit the map at any point in time. We now show how to leverage the representational capabilities of particle filters to address this more complex estimation problem.

A straightforward approach to partial map localization would be to estimate a robot's position both inside and outside the map. However, such an approach could be extremely inefficient since the area outside a partial map can be arbitrarily large. Fortunately, for the purpose of map merging, it is not necessary to reason about all possible locations outside the map. Instead, we are only interested in robot locations that are part of trajectories that overlap with the partial map (nonoverlapping trajectories correspond to cases in which the two maps do not overlap at all).

Similar to the application of Rao–Blackwellised particle filters for mobile robot mapping [5], [13], [23], one can consider a particle filter as recursively computing posterior probability distributions over robot trajectories. A particle filter represents such posterior distributions by sets $S_t = \{\langle x^{(i)}_{e_i:t}, w^{(i)}_t\rangle | i = 1, \ldots, N\}$ of N weighted samples distributed according to the posterior. Here, each $x^{(i)}_{e_i:t}$ is a trajectory, and the $w^{(i)}_t$ are nonnegative numerical factors called importance weights, which sum up to one. In our case, the trajectories have different lengths, as indicated by e_i, the time when trajectory i first entered the partial map. We proceed as shown in Table 1, in order to generate posteriors over such trajectories.

The algorithm takes as input the previous sample set along with the other robot's most recent control information and observation sent via wireless communication. Additionally, it requires the current partial map $\mathcal{M}_t$ and n_{t-1}, which is the probability that so far there was no overlap between the other robot's trajectory and the partial map. As soon as $\mathcal{M}_0$ is sufficiently large, the algorithm is started as follows. At $t = 0$, each trajectory consists of only one robot location x_0. In this degenerate case, only locations inside the partial map correspond to overlapping trajectories. Since there is no knowledge about the initial location of the robot, these "trajectories" x_0 are sampled uniformly throughout the partial map. n_0, the probability that the other robot initially is outside the partial map, is set according to an estimate of the ratio between the sizes of the partial map and the entire environment. Fig. 2(b) shows a partial map along with such a uniformly initialized sample set. The figure also shows the true robot location, which initially is outside the partial map.

Then, at each iteration of the particle filter, the trajectories are updated based on the following reasoning. At time t, a trajectory can overlap with the partial map if and only if it already overlapped at time $t-1$ or if the robot just entered the partial map for the first time. The first case is handled by Steps 3–5 of the algorithm. Here, each overlapping trajectory of the previous time step is extended using the motion information u_{t-1}. Then, in Steps 6–9, the algorithm generates N_ε locations that correspond to trajectories that enter the partial map for the first time. These trajectories do not contain locations prior to time t, since we are mostly interested in their locations inside the partial map (in fact, our efficient

Table 1 Outline of Particle Filter-Based Implementation of Partial Map Localization

```
1.  Inputs: $S_{t-1} = \{\langle x^{(i)}_{e_i:t-1}, w^{(i)}_{t-1}\rangle \mid i = 1,\ldots,N\}$, control information $u_{t-1}$, observation $z_t$, partial map $\mathcal{M}_t$,
        probability of non-overlapping trajectories $n_{t-1}$, probability of entering partial map $\varepsilon$, number of entry point samples $N_\varepsilon$
2.  $S_t := \emptyset$                                                   // Initialize
3.  for all samples $\langle x^{(i)}_{e_i:t-1}, w^{(i)}_{t-1}\rangle$ in $S_{t-1}$ do          // Extend existing trajectories
4.      sample $x^{(i)}_t$ from $p(x_t \mid x^{(i)}_{e_i:t-1}, u_{t-1})$           // Predict next position using motion
5.      $S_t := S_t \cup \{\langle x^{(i)}_{e_i:t}, w^{(i)}_{t-1}\rangle\}$                  // Insert into next set
6.  for $i := 1,\ldots,N_\varepsilon$ do                                   // Generate $N_\varepsilon$ new trajectories starting at entry points
7.      sample $x^{(i)}_t$ from an entry point into the map          // Entry points are given by transition from free space to unexplored
8.      $w^{(i)}_t = \frac{\varepsilon\, n_{t-1}}{N_\varepsilon}$                                    // Set weights accordingly
9.      $S_t := S_t \cup \{\langle x^{(i)}_t, w^{(i)}_t\rangle\}$                       // Insert into set
10. $n_t = (1-\varepsilon)\, n_{t-1}$                                       // Subtract fraction $\varepsilon$ migrated from non-overlapping trajectories into map
11. for all samples $\langle x^{(i)}_{e_i:t}, w^{(i)}_t\rangle$ in $S_t$ do                // Integrate observation into individual trajectories
12.     if $x^{(i)}_t \in \mathcal{M}_t$ then $w^{(i)}_t := w^{(i)}_t\, p(z_t \mid x^{(i)}_t)$        // Trajectories currently inside the map
                            else $w^{(i)}_t := w^{(i)}_t\, p(z_t \mid \text{outside})$      // Trajectories currently outside the map
13. $n_t = n_t\, p(z_t \mid \text{outside})$                                 // Update probability of non-overlapping trajectories
14. $\alpha_t^{-1} = n_t + \sum_{i=1,\ldots,N+N_\varepsilon} w^{(i)}_t$                        // Compute normalization factor
15. $n_t = \alpha_t n_t;\quad \forall i: w^{(i)}_t = \alpha_t w^{(i)}_t$                        // Normalize
16. resample samples in $S_t$ based on their weights                 // Samples $N$ trajectories
```

implementation only keeps track of the most recent location in each trajectory). The entry points in Step 7 are generated based on the assumption that the robot can only enter the partial map at its frontier cells. These cells are shown in Fig. 2(a), and the corresponding entry samples are indicated in the sample set shown in Fig. 2(c). The weights of these samples are set in Step 8 such that the combined weights of all new entry samples is $\varepsilon\ n_{t-1}$. The parameter ε represents the probability that the robot enters the map at any point in time given that it previously was outside. Step 10 adjusts the probability of nonoverlapping trajectories by subtracting the weights of trajectories that entered the map. At this point in time, the weights of all overlapping trajectories (extended and new ones) plus the probability n_t of nonoverlapping trajectories sum up to one (the only change is due to a shift of probability from nonoverlapping to just entering trajectories).

So far, we only considered robot motion, the most recent observation is integrated in Steps 11–14 of the algorithm. Here, we have to consider two cases depending on whether the location is inside or outside the partial map. Step 12 handles locations inside the partial map by multiplying the trajectory weight with the observation likelihood, which can be extracted from the map, exactly as in the regular robot localization [8], [9], [34]. Trajectories that overlap with the map but exited it at one point in time are weighted by the likelihood of observing the measurement outside the map (we will discuss our approach to computing this likelihood in the next section). The same likelihood is used to weight the probability n_t of nonoverlapping trajectories in Step 13. The normalization

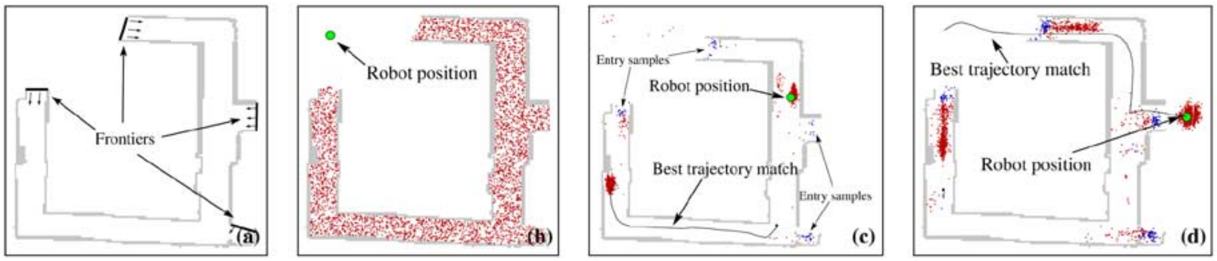


Fig. 2. ***Partial map localization. (a) Particle filter generates entry point samples (blue) along arcs at each frontier. (b)–(d) Sample sets at different points in time. Pictures also show true robot location, entry point samples, and most likely hypothesis for other robot's position along with trajectory of this hypothesis. (b) Initially, samples are spread uniformly throughout partial map. (c) After only short overlap, most likely trajectory does not match true path of robot. (d) Robot exits map and most likely particle is at correct position.***

factor is determined in the following step and multiplied into n_t and all weights in Step 15. The final step samples N trajectories from the weighted samples and sets the weights such that they sum up to $1 - n_t$.

It can be shown that each iteration of this particle filter generates samples that are distributed according to the posterior over trajectories that overlapped with the partial map at some point in time. If the size of the partial map increases during this process, we move the entry point frontiers accordingly.

B. Predictive Model for Observations Outside the Partial Map

A key quantity estimated by the particle filter algorithm is n_t, which is the probability of whether or not there is an overlap between the partial map and the other robot's path. This quantity is crucial to assess the weight $p(c_{pq})$ of a map merge hypothesis used by the coordination algorithm in (3). In order to determine n_t, it is necessary to estimate the likelihood of sensor measurements outside the partial map (Steps 12 and 13). Unfortunately, an accurate estimate of this likelihood is not possible, since the robots do not know which measurements to expect outside the explored area. One solution to this problem is to use a fixed likelihood for all observations z_t made at locations outside the partial map. However, such an approach ignores valuable information about the structure of an environment and results in brittle estimates of map overlaps [30].

To acquire $p(z_t|\text{outside})$, the likelihood of observing z_t in unexplored areas, we developed a structural model of indoor environments that can be used to predict the observations made by a robot. An in-depth discussion of this approach is beyond the scope of this paper, thus we refer the reader to [10] and [30] for more details. In a nutshell, the structural model is a hidden Markov model that generates sequences of views observed by a robot when navigating through an environment. In our approach, we extract discrete views v from laser range scans. These views roughly correspond to map patches such as hallways, openings, rooms, etc. At every update of the particle filter, the next view is predicted based on the previous view and the view transition probability. Both view and transition probabilities are estimated during exploration via so-called Dirichlet hyper-parameters. More specifically, let v_t denote the random variable over views observed at time t. It can be shown [10] that the predictive distribution for observing view i at time t, given that the robot just observed view j, is given by

$$p(v_t = i | v_{t-1} = j, \boldsymbol{\alpha}_j, \mathbf{f}_{|j}) = \frac{f_{i|j} + \alpha_{i|j}}{\sum_{i'} f_{i'|j} + \alpha_{i'j}}. \tag{4}$$

Here, $\mathbf{f}_{|j}$ is the number of times view j has already been observed in this environment, and $\boldsymbol{\alpha}_j$ is a Dirichlet prior count learned from previous environments. Accordingly, $\boldsymbol{f}_{i|j}$ and $\boldsymbol{\alpha}_{i_j}$ are the view transitions observed in this environment and given as prior counts, respectively. The Dirichlet prior counts are learned using a hierarchical Bayesian approach. At each iteration of the particle filter, (4) is used to estimate $p(z_t|\text{outside})$ in Steps 12 and 13 of the algorithm. In [10] and [30], we showed that this predictive model results in significantly better estimates of whether or not the maps of two robots overlap. By updating the probabilities of the hidden Markov model (HMM) as the robots explore an environment, our model achieves an additional improvement in predictive quality [10].

At each iteration of the particle filter, hypotheses for the location of a robot are extracted from the sample set and then used by the decision-theoretic coordination technique described in Section II-B. Once the coordination approach considers a hypothesis valuable enough, it verifies this hypothesis by assigning it to a robot. If this robot detects the other robot at the hypothesized position, its data can be merged into the cluster map, as described next. If, however, the hypothesis turns out to be incorrect, then the particle filter naturally incorporates such information by giving the samples at the wrongly hypothesized location extremely low weights. The low weights result in the removal of these particles in the next resampling step, thereby increasing the probability of alternative hypotheses.

IV. MULTIROBOT MAP MERGING

We will now describe how to build a consistent map from data collected by multiple robots.

A. SLAM Paradigms and Local Maps

The key problem in mobile robot mapping is caused by the uncertainty in a robot's position as it explores an environment. This position uncertainty has to be considered when generating a map from the observations made by the robot. It is this connection between robot position and map uncertainty that makes the simultaneous localization and mapping (SLAM) problem computationally demanding [4], [33]. Over the last several years, various research groups have developed efficient solutions to the SLAM problem. These techniques range from splitting maps into submaps [27], to thin junction-tree approximations [28], to sparse extended information filters [33], to Rao–Blackwellised particle filters [5], [13], [23], [25], to graph structures modeling spatial constraints [12], [18], [22]. In this project, we build on the latter class of techniques, which are appropriate because they can be made to be independent of the coordinate system in which the constraints are expressed [19], an obvious advantage when combining local maps that have different coordinate systems. Here, we only provide an intuitive description of our approach, more details can be found in [12], [19], and [22]; a good exposition of general constraint graphs can be found in the Graph-SLAM algorithm [34].

B. Local Constraints and Optimization

The key to combining information from multiple local maps is to form probabilistic constraints that are invariant to rigid transformations. Such constraints can be combined directly, because they are not tied to any particular local map coordinate system. Constraints are generated from four sources:

1) odometry between successive poses;
2) scan-matching between nearby poses;
3) loop closure, when scans from two historically distant poses are matched;
4) colocations between poses in partial maps.

Formally, constraints in our system are measurement equations between pairs of poses. For two poses p_0 and p_1, the equation is the difference between the two poses $f(p_1, p_0) = p_1 - p_0$. The measured distance is given by a difference d_{01} and its covariance C_{01}. For example, using odometry the difference comes from the measured wheel movement, and the covariance from a model of the motion errors. The other source of constraints comes from scan matching between poses (cases 2–4). As in the case of odometry, the output of the scan match is an estimated difference between the poses, and a covariance of the estimate. Thus, all of the constraints we are working with can be put into the form of measurements of the difference between two poses.

Unfortunately, constraints that are pure pose differences are not local—they can vary depending on the global location of the poses. Consider the two poses located at $p_0 = (0, 0, 0)$ and $p_1 = (0, 1, 0)$. Their difference is $d_{01} = (0, 1, 0)$. Now rotate the coordinate system 90° so that $p_0 = (0, 0, pi/2)$ and $p_1 = (1, 0, pi/2)$. Obviously, their difference d_{01} in the new coordinate system will change. When we put together constraints from different partial maps, each of which has its own coordinate system, the constraints are not comparable. Our solution is to always express constraints in a form which is invariant to a rigid transformation of the pose coordinates. Instead of using pose differences in the global system, we express the difference in the coordinate system of p_0, that is, as if p_0 were at (0,0,0). We write ${}^0(p_1 - p_0)$ for this relative pose difference; it is easy to verify that it is invariant to any rigid transformation of the global coordinates.

Given a set of local pose constraints, the maximum likelihood solution is found by minimizing the covariance-based errors. For each constraint k, define

$$\epsilon_k = d_{ij} - {}^i(p_j - p_i). \tag{5}$$

The total covariance-based squared error (also called the squared Mahalanobis distance) is given by

$$E = \sum_k \epsilon_k^T C_k^{-1} \epsilon_k. \tag{6}$$

Any particular set of values for the pose variables will yield a value for E. Finding the values that minimize E is a nonlinear optimization problem (the constraints are nonlinear because of the angular dependencies). Given an initial solution that is close enough to the optimal solution, there are efficient methods for solving this problem, most notably conjugate gradient descent [12], [18], [22], [34]. In practice, these methods work very well and can solve systems of (for example) 1000 poses in under 1 s.

C. Map-Merging Examples

The constraint graph is ideal for integrating map information with uncertain alignment. In the case of odometry and local scan matches, the system looks at just a small local neighborhood to enforce consistency [12], which can be done in constant time. The more interesting cases are enforcing global consistency: loop closure in a local map and partial map merging. In loop closure, a robot is building a local map using its own scans and the scans of any colocated robots. At some point, the robot returns to a position it has previously visited, but accumulated error causes it to be misaligned [Fig. 3(a)]. Here, the robot has traversed an interrupted loop, going out of the top of the figure before coming back. Once scan matching establishes links with poses at the beginning of the loop, additional constraints can be added to the graph. Based on these constraints, the mapping algorithm determines the optimal position for all scan locations by maximizing the posterior probability of all constraints in the graph, using (6). In practice, the initial solution established by enforcing local constraints gives a "close enough" solution to start the minimization process. In Fig. 3(b),

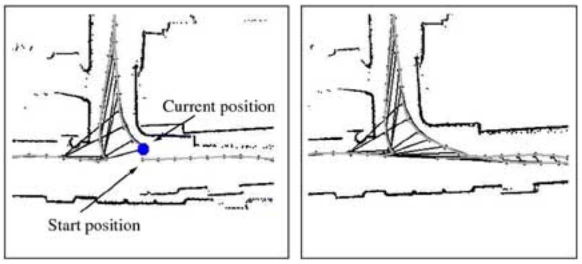


Fig. 3. ***Pose constraints (a) before and (b) after linking start and end of loop. Minimization of constraints after robot returned into hallway to right results in consistent scan locations. Robot trajectory is shown in gray, spatial constraints as thin black lines attached to trajectory.***

scan matching has established links with poses at the beginning of the loop, resulting in a consistent map after minimization of the constraint system. Because the optimization is efficient, it can be performed online as the robot explores an environment, causing no more than a second or so of hesitation as consistency is enforced.

The constraint representation naturally facilitates the merging of partial maps built by different robots. For example, the upper panels of Fig. 4 show three partial maps built by three robots. Suppose we can link the pose marked "o" in the left map to the pose marked "o" in the middle map and the poses marked "x" in the middle and the right maps. Then, we can move the three maps together to register them in the same metric space. This is done by taking each constraint in the middle map and adding it to the constraint graph underlying the left map, just as if all scans were collected by a single robot. In addition, we generate an initial solution as input to the global optimization, by transforming all the poses in the middle map, and making a rigid transformation so that they line up with the colocated pose at its correct position. At this point, although the maps are aligned correctly for the colocated poses, they can differ on poses that are distant from this point—see the lower right panel in Fig. 4. An additional "zippering" process is performed in which all the poses that are now close in the colocated two partial maps are scan matched for additional constraints. By consolidating the poses into spatial buckets, this process can take place in order N, the number of poses in the partial map. Optimization of (6) yields a globally consistent map. Finally, the scans observed by the third robot are added to this map, using the same process. The occupancy grid map resulting from optimizing the global constraint graph is shown in the lower left panel in Fig. 4.

Abstractly, the zippering process lets us take any partial maps produced by any robots and put them together, once a common location (colocation) between their trajectories has been identified. In our system, colocation information is estimated by the particle filter described in Section III and verified via actively triggered robot detections, as described in Section II. Our map-merging technique is transitive in the sense that if robot A knows robot B's location inside its partial map and robot B knows the location of robot C inside its partial map, then it is possible

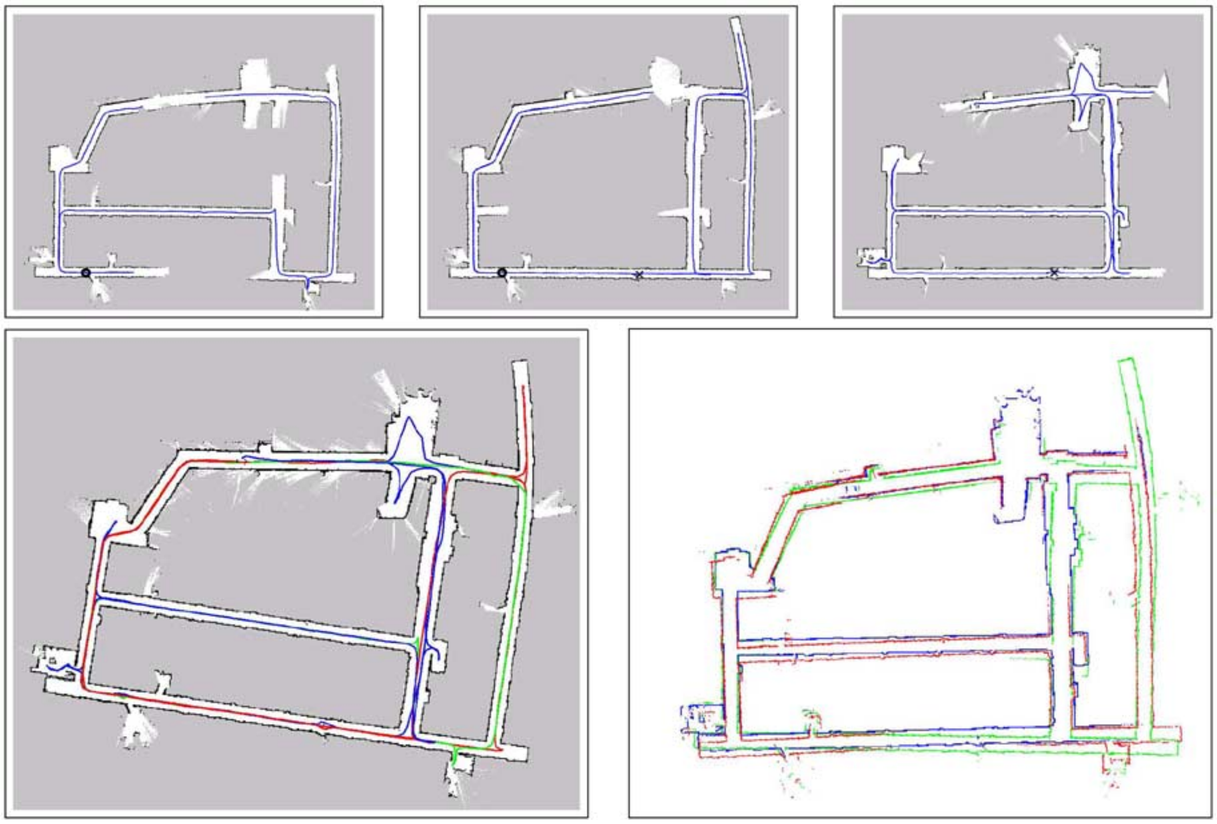


Fig. 4. *Upper panels: partial maps built by three robots in the UW Allen Center. "o"s and "x"s provide connection points between left and middle map, and middle and right map, respectively. Lower panels: left picture shows map generated from three partial maps by optimization of global constraint graph generated by "zippering" maps together at connection points. (right) Map generated by simply overlaying partial maps, without any additional global optimization (only laser scans are shown for clarity).*

to consistently merge C's map into robot A's map (possibly after merging B's map into A's map). The reader may notice that merging maps in different orders might lead to slightly different maps, which is due to the approximations performed by our approach (sequentially adding spatial constraints might result in different constraint systems). In practice, however, we found this approach to map merging highly reliable.

V. EXPERIMENTAL EVALUATION

We will now describe the evaluation of our exploration system. Additional aspects of the approach are evaluated in [10], [16], [18], and [30].

A. Implementation Details

1) Coordination and Mapping: We implemented the decision-theoretic coordination technique described in Section II-B. Maps are represented compactly as sets of laser range scans annotated with robot poses and probabilistic links (scans are recorded only every 50 cm of translation or 30 degrees of rotation). Within an exploration cluster, each robot integrates its observations into its own map and broadcasts the information to the other robots. While most of the other robots only store this data, the team leader of the cluster integrates all the sensor information it receives. Thus, the team leader, which is chosen as the robot with the smallest ID, has a complete and consistent map representing the data collected by all robots in the cluster. This map is used to coordinate the robots in the cluster. Whenever two clusters meet and merge their maps, the team leader with the smaller ID becomes the leader of the new exploration cluster. Frequently, the team leader broadcasts the map to the other robots, in order to guarantee consistency. This data can be sent very compactly, since only updated robot poses and links have to be transmitted (scans are already stored by the other robots). The most complex broadcast follows whenever a robot closes a loop, since the optimization of the constraint system modifies all robot poses in a map (Section IV). In practice, broadcasting even this information typically involves sending only several kilobytes of data, which is well below the capacity of typical 802.11 wireless communication capacity and can be done in a fraction of a second.

A crucial situation occurs when a robot moves into the communication range of a robot from another, possible single-robot, cluster. At this point in time, the robots exchange all their sensor and motion information and start estimating their relative locations using the particle filter discussed in Section III. Our current system allocates this task to the team leader. The other robots in the team do not generate hypotheses. In the worst case, if all robots are in single-robot clusters and within communication range, the number of particle filters run on each robot can be as high as the total number of robots minus one. In simulation experiments, we found this simple approach to work efficiently enough for up to six robots. However, it does not scale to very large teams of robots and more thought must be put into more intelligent allocation of computation tasks.

2) Dealing With Limited Communication: Our exploration system achieves robustness to communication loss by enabling every robot to explore the environment on its own. Whenever a robot in an exploration cluster reaches an assigned goal point, it keeps on exploring based on its own map until it receives a new goal point. Thus, if a robot moves outside the communication range of its cluster, it automatically keeps on building its own map until it gets back into communication range. After getting back into communication, robots exchange all the relevant data that was lost. Such a "sync" operation only involves the communication of rather small data sets and can typically be done in less than a second. Our approach is also robust to loss of the team leader, since any other robot in the cluster can explore on its own or take over the team leader role. In the extreme, if none of the robots can communicate with each other, each robot will explore the environment independently of the other robots. The result will still be a complete map, only built less efficiently.

We added hand shaking and various timeouts to the decision making in order to make active hypothesis verification robust to loss of communication. This implementation task turned out to be rather tedious since it required extensive testing of the system in order to determine all situations in which one robot might leave the communication range of another robot. As an example of our approach, when a robot sends a "Meet" signal to another robot for which it has a good location hypothesis, it waits for an acknowledgment of this signal. If the acknowledgment is not received after several seconds, the robot keeps on exploring and reconsiders a meeting only after an additional timeout and the other robot is back in communication.

B. Predictive Model for Estimating Relative Locations

As described in Section III, our system relies on particle filters to estimate the relative positions of robots. In order to estimate whether or not the maps of two robots overlap, we compare the likelihood of measurements z inside a partial map with the likelihood of observing z outside the map, denoted $p(z|\text{outside})$. To estimate the outside likelihood, we developed a hierarchical Bayesian technique that learns priors from previously explored environments (see Section III-B).

To evaluate the suitability of this approach for partial map merging, we generated 15 partial maps from five different environments and estimated the location of a robot relative to these maps using the approach described in Section III. For each partial map, we took several sensor logs collected in the same environment. The sensor logs were chosen randomly and some of them had no overlap with the corresponding partial map at all. For each map–trajectory pair we proceeded as follows. At each iteration of the particle filter, we determined the most likely hypothesis for the robot's location. If the probability of this hypothesis exceeded a certain threshold θ, then this hypothesis was considered valid. For each threshold θ, precision measures the fraction of correct valid hypotheses, i.e., hypotheses above the threshold. Correctness is tested by comparing the position of the hypothesis to a ground truth estimate computed offline. To determine recall, we first checked at what times the robot was in the partial map. Recall, then, measures the fraction of this time for which the approach generated a correct hypothesis, i.e., at the correct position and with probability above the threshold θ. We compared our approach to an alternative method that uses a fixed likelihood $p(z|\text{outside})$ for locations outside the partial map (this corresponds to virtually all existing mapping techniques).

The precision–recall tradeoffs for different thresholds θ are shown in Fig. 5(a) (each point represents a different threshold). The solid line represents the results obtained with our approach and the dashed lines are results for the fixed approach using different likelihoods $p(z|\text{outside})$ for measurements outside the maps (data points are omitted for clarity). The graphs clearly show the superior performance of our approach. It achieves 26% higher precision than the best likelihood value for the alternative method. Note that high precision values are more important than high recalls since low precision results in wrong active colocation decisions, while low recall only delays the map-merging process. Note also that one cannot expect very high recall values since a robot has to be in the partial map for a certain duration before a valid hypothesis can be generated.

To evaluate the predictive quality of our approach, we used sequences of data collected in three environments. At each iteration (after approximately 2 m of robot motion), we computed the likelihood of the next view in the data log given the view prediction obtained by our approach. The predictive quality is then determined by accumulating the logarithm of the measurement likelihoods over time. Fig. 5(b) shows the results for alternative techniques, averaged over 45 data sequences. The solid line represents the results for our approach, i.e., using (4) to predict the next view. The dashed line gives the results using our approach, but without updating the model, i.e., only the Dirichlet prior counts learned from other maps are used [$\boldsymbol{f}_{i|j}$ and $\boldsymbol{f}_{i'|j}$ are set to zero in (4)]. Even though the (logarithmic) difference between these top two graphs seems small, the average likelihood of a complete sensor sequence using our adaptive approach is approximately 360 times as high as with the prior only approach. This indicates that it is important to update the predictive model using observations obtained in the new environment. The dotted line in Fig. 5(b) shows the result if we predict views using the frequency counts of view transitions observed in the other maps. These predictions are clearly inferior to those of the Dirichlet prior learned with the hierarchical Bayesian approach (dashed line), which shows that our learning method significantly improves the performance over straightforward transition frequency counting. Finally, the dashed–dotted line gives the result based on frequency counts of individual views, i.e., without considering transitions between views. This graph demonstrates that considering the connectivity of environments is superior to predicting views simply based on their frequency. These graphs are averages over different environments. We found our approach to yield much stronger improvements in predictable environments such as typical office buildings.

C. Fort A. P. Hill Evaluation

Our overall exploration and mapping system was evaluated thoroughly as part of the CentiBots team within

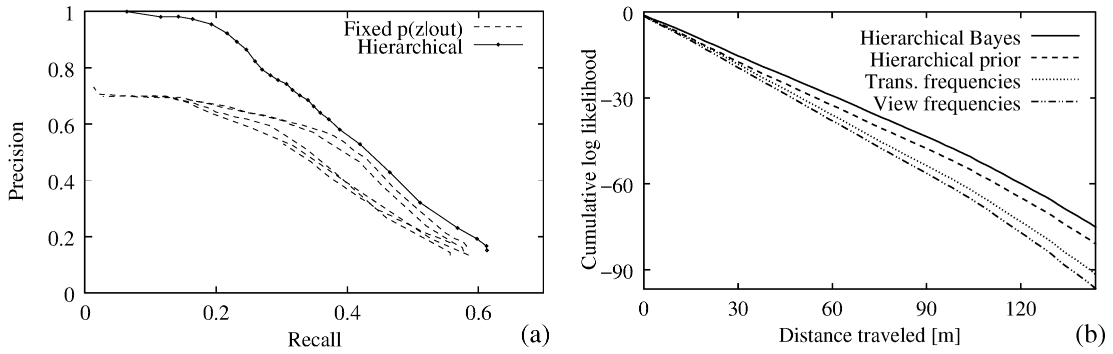


Fig. 5. ***(a) Precision versus recall. Each point represents average over 375 pairs of partial maps and trajectories. Each curve shows tradeoff for different thresholds θ (0.05–0.99). Dashed lines indicate results obtained with different fixed values for $p(z|\text{outside})$ and solid line represents results for our approach. (b) Predictive likelihood of different approaches averaged over 45 data sequences in three environments.***

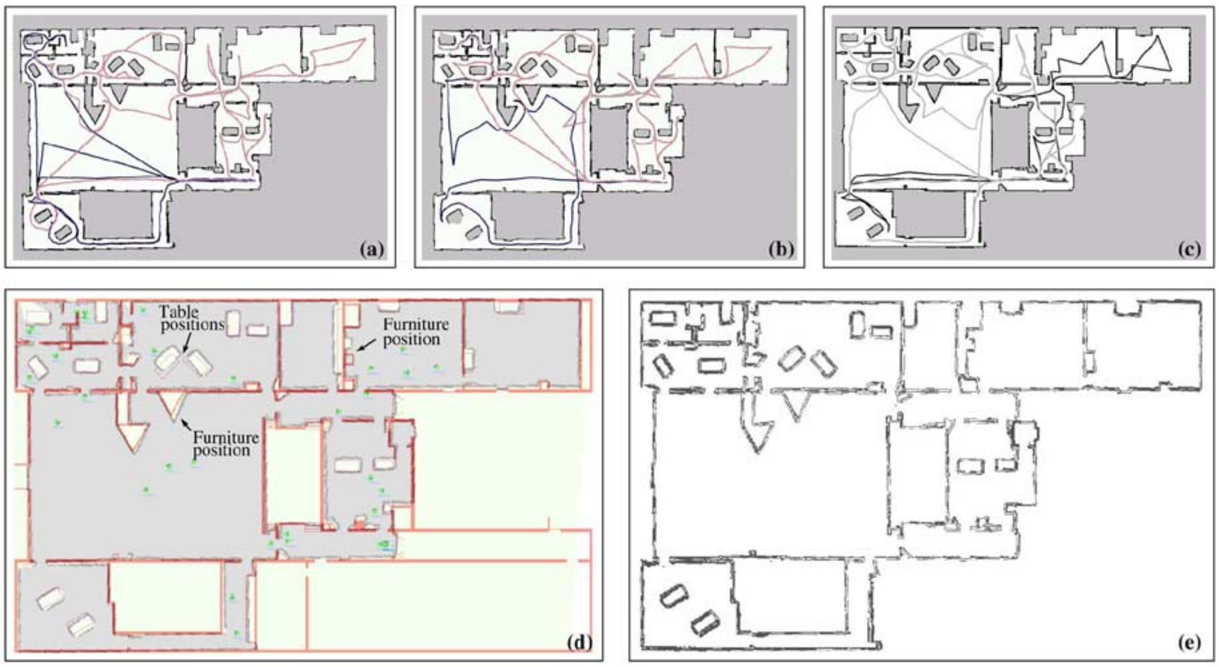


Fig. 6. *(a)–(c) Maps built during three autonomous exploration runs. Maps look almost identical, even though they were built under very different circumstances. Similarity between maps illustrates robustness of system and supports our belief that these maps are more accurate than hand-built map. (d) Map overlayed with ground truth CAD model of building. CAD model was generated by manually measuring locations and extensions of rooms and objects. (e) Map generated from overlaying three maps shown in (a)–(c). White pixels indicate locations at which all three maps agree, black pixels show disagreement on occupancy.*

the Defense Advanced Research Project Agency SDR project. The SDR project was unique in having an experimental validation conducted by an outside group. For a week in January 2004, the CentiBots were tested at a 650-m^2 building in Ft. A.P. Hill, Virginia. We were tested under controlled conditions, with a single operator in charge of the robot teams. All computation was performed using state-of-the-art laptops onboard the robots. The evaluation criteria for mapping included time to create a map, topological accuracy, and percent of area mapped. Ground truth for mapping was given by a manually constructed map [Fig. 6(d)]. Extensive software tuning was circumvented by limiting access to only half of the experimental area during test runs. Even the developer team was not allowed to inspect the complete environment before the robots were deployed.

The results for five official mapping runs are summarized in Table 2. In all runs, the robots were able to autonomously generate a highly accurate map of the environment. The average mapping time for single-robot exploration was 24 min; this time was reduced to 18 min when using two and 15 min when using three robots. It should be noted that the robots frequently lost communication with the overall control center and with other robots. In such situations, the robots explored on their own and combined their sensor data as soon as they were in contact again.

Table 2 Exploration Runs During Ft. A. P. Hill Evaluation

Run	# Mapping robots	Mapping Time	Map area
1	1	22 min	96%
2	1	26 min	97%
3	2	17 min	95%
4	2	19 min	96%
5	3	15 min	97%

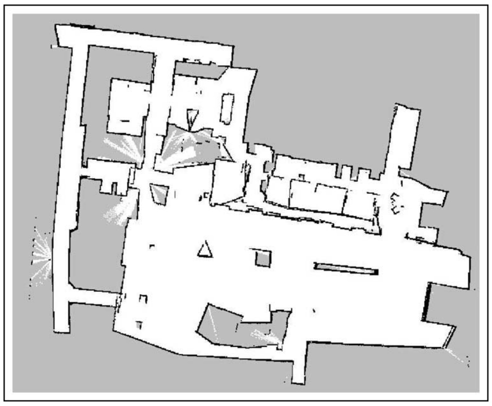


Fig. 7. *Map used for simulation experiments.*

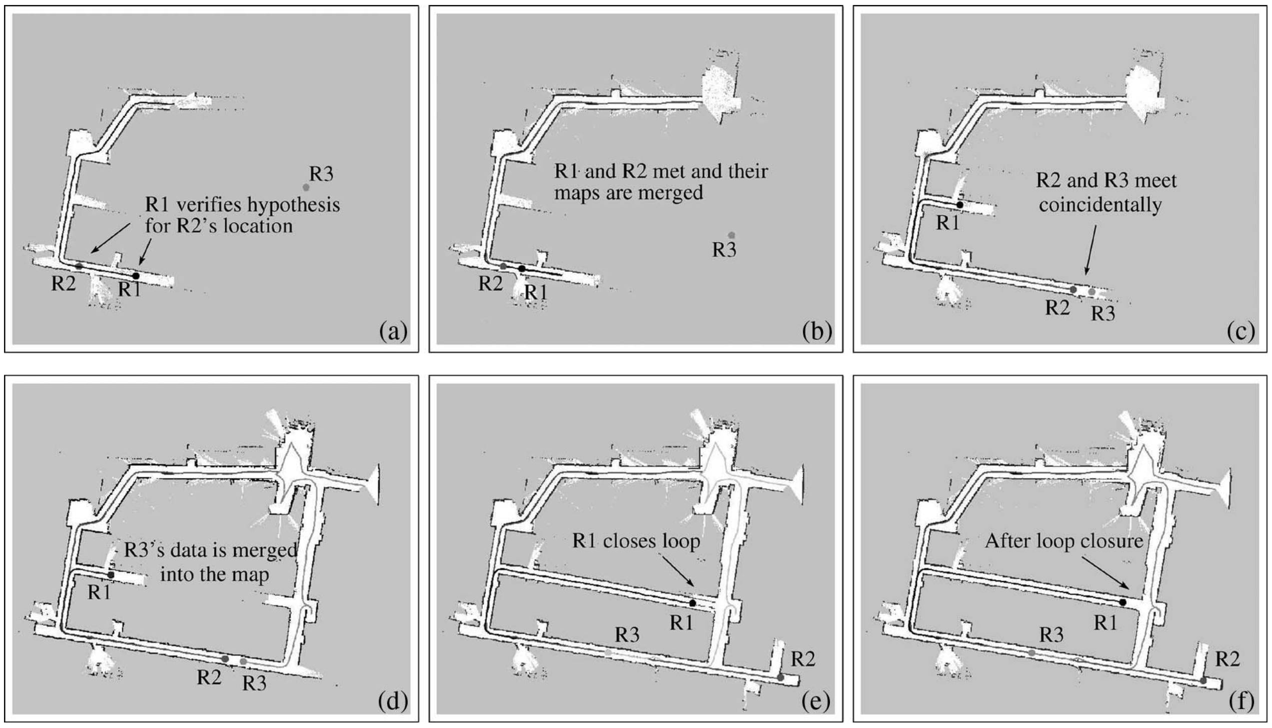

R3
R1 verifies hypothesis for R2's location
R2
R1
(a)
R1 and R2 met and their maps are merged
R3
R2
R1
(b)
R2 and R3 meet coincidentally
R1
R2
R3
(c)
R3's data is merged into the map
R1
R2
R3
(d)
R1 closes loop
R3
R1
R2
(e)
After loop closure
R3
R1
R2
(f)

Fig. 8. ***Sequence of partial maps generated during exploration with three robots. Shown is only map of robot R1 along with locations of other two robots. In this experiment, robots do not know their relative start locations. (a) R2 is in R1's map and R1 has a high probability hypothesis for R2's location. R1 sends R2 a "Stop" command and decides to verify this hypothesis. (b) After R1 meets R2 at hypothesized location, robots merge their maps. They now coordinate their exploration. (c) R2 and R3 meet incidentally and (d) merge R3s data into map. (e) R1 closes loop. (f) Since all data is integrated into global constraint graph, R1 is able to correct odometry error. Final map of this run is shown in lower left panel of Fig. 4.***

In addition to the robustness of our system, an important result of this evaluation was the fact that the maps built by our approach were more accurate than those built manually by the evaluation team. Fig. 6(d) shows one of our maps overlayed with the "ground truth" map. As can be seen, the two maps do not match perfectly (see, for instance, the two tables in the upper middle room). Three maps built by our robots in three different evaluation runs are shown in Fig. 6(a)–(c). These maps look virtually identical, even though they were built independently of each other, using different trajectories of the robots. To better illustrate the similarity of these three maps, we overlayed them. Fig. 6(e) shows the pixels at which the overlayed occupancy grid maps are not identical. As can be seen, the maps almost perfectly line up. Mismatches are only along the obstacles, which is mostly due to limited resolution of the maps.

D. UW Allen Center Evaluations

We performed additional evaluation runs with three robots in the UW Allen Center. These runs confirmed the reliability of our system. Fig. 8 illustrates one of these runs. An animation of this run can be found.[1]

[1]http://www.cs.washington.edu/robotics/projects/centibots.

In order to evaluate the benefits of active colocation, we performed several simulation runs involving three robots. To do so, we used the Saphira robot simulator along with a map of the first floor of the Allen Center (see Fig. 7). The Saphira simulator accurately models robots including noise in motion and sensing. A larger environment was simulated by limiting the velocity of robots to 20 cm/s and the range of detections to 1.5 m.

Table 3 summarizes the results of 12 exploration runs, six of which were performed using our active colocation approach (numbers are mean times along with 95% confidence intervals). The other six runs merged maps only when robots met coincidentally, similar to the approach discussed in [14]. Our map-merging technique generated accurate, consistent maps in all 12 runs. As can be seen, actively verifying relative location hypotheses significantly reduces the overall exploration time. The third and fourth column of the table indicate why our

Table 3 Exploration With and Without Active Colocation

Approach	Mapping time [min]	First meeting [min]	Second Meeting [min]
Passive	35.2±3.8	15.0±4.1	25.2±7.9
Active	26.3±2.4	6.2±1.3	16.3±3.8

active approach is more efficient than its passive counterpart. The third column provides the time until the first two robots are able to merge their maps, and the fourth column gives the time until the third robot joins the exploration cluster. As can be seen, by actively verifying hypotheses, the robots are able to merge their maps earlier, which results in improved coordination between their exploration strategies.

We performed further simulation runs in this environment using six robots, which resulted in faster exploration of 22.8 ± 4.5 min. All these runs resulted in globally consistent maps. Furthermore, these runs involved active colocation between exploration clusters of more than one robot each.

VI. CONCLUSION

We presented a distributed approach to mobile robot mapping and exploration. The system enables teams of robots to efficiently explore environments from different unknown locations. The robots initially explore on their own until they can communicate with other robots. Once they can exchange sensor information with other robots, they estimate their relative locations using an adapted particle filter. In order to estimate whether or not the partial maps of two robots overlap, the filter incorporates a hidden Markov model that predicts observations outside the explored area. The parameters of the model are learned from previously explored environments using a hierarchical Bayesian approach. During exploration, the robots update their predictive models based on observations in the new environment. Our experiments indicate that this approach supports map-merging decisions significantly better than alternative techniques.

The estimation of relative positions is integrated seamlessly into a decision-theoretic multirobot coordination strategy. In order to overcome the risk of false-positive map matches, the robots actively verify location hypotheses using a rendezvous strategy. If the robots meet at the meeting point, they know their relative locations and can combine their data into a shared map. Mapping and map merging uses a SLAM technique that models uncertainty by local probabilistic constraints between the locations of laser range-scans. Shared maps are used to coordinate the robots and to estimate the location of other robots.

Our mapping and exploration system was evaluated under some of the toughest real-world conditions yet imposed on a robotics project. Prior to the evaluation, the developer team was allowed to test their robots only in one half of the environment. The other half was not accessible during testing. During the evaluation runs, the robots had to rely on their own *ad hoc* wireless network to exchange information. The robots successfully explored the environment in all four official evaluation runs. All maps generated during these runs were virtually identical, indicating the high accuracy and robustness of our system. ■

Acknowledgment

The authors would like to thank D. Gage and DARPA for support under the Software for Distributed Robotics Program. They would also like to thank the evaluation team, E. Krotkov and D. Hackett, for their extraordinary efforts in designing and running the demonstration. Other members of the CentiBots team (C. Ortiz, A. Agno, M. Eriksen, B. Morisset, and R. Vincent) were extremely helpful in the deployment and testing of our approach.

REFERENCES

[1] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2000.

[2] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 376–386, Jun. 2005.

[3] G. Dedeoglu and G. S. Sukhatme, "Landmark-based matching algorithm for cooperative mapping by autonomous robots," in *Proc. 5th Int. Symp. Distributed Autonomous Robotic Systems (DARS)*, 2000.

[4] M. W. M. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Trans. Robot. Automat.*, vol. 17, no. 3, Jun. 2001.

[5] A. Eliazar and R. Parr, "DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2003.

[6] J. W. Fenwick, P. M. Newman, and J. J. Leonard, "Cooperative concurrent mapping and localization," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2002.

[7] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *Int. J. Robot. Res. (IJRR)*, vol. 22, no. 12, 2003.

[8] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo Localization: Efficient position estimation for mobile robots," in *Proc. Nat. Conf. Artificial Intelligence (AAAI)*, 1999.

[9] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *J. Artif. Intell. Res. (JAIR)*, vol. 11, pp. 391–427, 1999.

[10] D. Fox, J. Ko, K. Konolige, and B. Stewart, "A hierarchical Bayesian approach to mobile robot map structure learning," in *Robotics Research: The Eleventh Int. Symp., Springer Tracts in Advanced Robotics (STAR)*, P. Dario and R. Chatila, Eds. New York: Springer Verlag, 2005.

[11] B. Gerkey and M. Mataric, "Multi-robot task allocation: Analyzing the complexity and optimality of key architectures," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2003.

[12] J. S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *Proc. IEEE Int. Symp. Computational Intelligence Robotics Automation (CIRA)*, 1999.

[13] D. Hähnel, W. Burgard, D. Fox, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems (IROS)*, 2003.

[14] A. Howard, L. E. Parker, and G. S. Sukhatme, "The SDR experience: Experiments with a large-scale heterogenious mobile robot team," in *Proc. Int. Symp. Experimental Robotics (ISER)*, 2004.

[15] P. Jensfelt, O. Wijk, D. Austin, and M. Andersson, "Feature based condensation for mobile robot localization," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2000.

[16] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai, "A practical, decision-theoretic approach to multi-robot mapping and exploration," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems (IROS)*, 2003.

[17] S. Koenig, C. Tovey, and W. Halliburton, "Greedy mapping of terrain," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2001.

[18] K. Konolige, "Large-scale map making," in *Proc. Nat. Conf. Artificial Intelligence (AAAI)*, 2004.

[19] ——, "SLAM via variable reduction from constraint maps *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2005.

[20] K. Konolige, D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai, J. Ko, B. Morisset, D. Schulz, B. Stewart, and R. Vincent, "Centibots: Very large scale distributed robotic teams," in *Experimental Robotics: 9th Int. Symp., Springer Tracts in Advanced Robotics (STAR)*, M. Ang and O. Khatib, Eds. New York: Springer Verlag, 2005.

[21] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2000.

[22] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Auton. Robot*, vol. 4, pp. 333–349, 1997.

[23] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: Afactored solution to the simultaneous localization and mapping problem," in *Proc. Nat. Conf. Artificial Intelligence (AAAI)*, 2002.

[24] S. Moorehead, "Autonomous surface exploration for mobile robots," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 2001.

[25] K. Murphy, "Bayesian map learning in dynamic environments," in *Advances Neural Information Processing Systems (NIPS)*, 1999.

[26] E. Nettleton, S. Thrun, and H. Durrant-Whyte, "Decentralised SLAM with low-bandwith communications for teams of airborne vehicles," in *Proc. Int. Conf. Field Service Robotics*, 2003.

[27] P. Newman and J. J. Leonard, "Consistent convergent constant time SLAM," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2003.

[28] M. A. Paskin, "Thin junction tree filters for simultaneous localization and mapping," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2003.

[29] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes, "Coordination for multi-robot exploration and mapping," in *Proc. Nat. Conf. Artificial Intelligence (AAAI)*, 2000.

[30] B. Stewart, J. Ko, D. Fox, and K. Konolige, "The revisiting problem in mobile robot map building: A hierarchical Bayesian approach," in *Proc. Conf. Uncertainty Artificial Intelligence (UAI)*, 2003.

[31] A. Stroupe, R. Ravichandran, and T. Balch, "Value-based action selection for exploration and dynamic target observation with robot teams," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2004.

[32] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *Int. J. Robot. Res.*, vol. 20, no. 5, 2001.

[33] ——, "Robotic mapping: A survey *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Nebel, Eds., San Francisco, CA: Morgan Kaufmann, 2002.

[34] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[35] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte, "Towards multi-vehicle simultaneous localisation and mapping," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2002.

[36] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proc. Second Int. Conf. Autonomous Agents*, 1998.

[37] R. Zlot, A. Stentz, M. Bernardine Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proc. IEEE Int. Conf. Robotics Automation (ICRA)*, 2002.

ABOUT THE AUTHORS

Dieter Fox received the Ph.D. degree from the University of Bonn, Bonn, Germany.

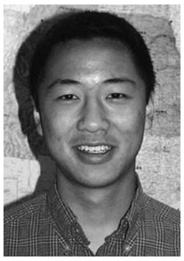
He is an Associate Professor and Director of the Robotics and State Estimation Lab in the Computer Science and Engineering Department, University of Washington (UW), Seattle. Before joining UW, he spent two years as a Postdoctoral Researcher at the CMU Robot Learning Lab. His research interests include artificial intelligence and robotics, with a focus on probabilistic state estimation and machine learning. His most recent work is in mobile robot mapping and sensor-based activity recognition. He is coauthor of over 100 technical papers and the book *Probabilistic Robotics* (MIT Press, 2005).

Dr. Fox is on the editorial boards of the IEEE Transactions on Robotics and the *Journal of Artificial Intelligence Research*. He has received various awards, including an NSF CAREER award and best paper awards at robotics (IROS-98, ICRA-00, RoboCup-04) and artificial intelligence conferences (AAAI-98, AAAI-04).

Kurt Konolige received the Ph.D. degree in computer science from Stanford University, Stanford, CA, in 1984.

He is a Senior Computer Scientist at the Artificial Intelligence Center of SRI International, a Consulting Professor of computer science at Stanford University, and a Fellow of AAAI. His recent research interests have included real-time perception and navigation for mobile robots. He teaches a course in mobile robotics at Stanford University and codeveloped the Pioneer and AmigoBot robot line and the Saphira robot control architecture. Recent projects focus on visual perception, mapping, and navigation using probabilistic techniques. He has been an Invited Lecturer at universities and institutions in many different countries. He has authored or coauthored over 100 scientific publications and three books.

Dr. Konolige is or has been on the editorial board of various academic publications, including *Fundamenta Informaticae, Journal of Applied Non-Classical Logics, International Journal of Applied Intelligence, Artificial Intelligence Journal*, and the *Journal of Artificial Intelligence Research*. He received Best Paper Awards at the 1995 IJCAI conference and the 1998 IROS conference.



Jonathan Ko received the B.S. degree from the University of California, Berkeley, in 2000, and the M.S. degree in robotics from the University of Washington, Seattle, in 2003. He is pursuing the Ph.D. degree from the Department of Computer Science and Engineering, University of Washington.

His research interests include exploration, planning, and multirobot systems.

Benson Limketkai received the B.S. degree in computer science from the University of California, Berkeley, and the M.S. degree from Stanford University, Stanford, CA. He is currently pursuing the Ph.D. degree from the University of Washington, Seattle, where his advisor is D. Fox.

He interned at SRI International for a summer and worked on the Centibots project for the following year and the subsequent summer.

Dirk Schulz received the Ph.D. degree in computer science from the University of Bonn, Bonn, Germany, in 2002.

He is a Postdoctoral Researcher in the Department of Computer Science III, University of Bonn. His research interests include probabilistic sensor fusion and state estimation technisques with applications in mobile robotics and intelligent environments.

Benjamin Stewart was born and raised in Ellicott City, MD. He received the B.S. degree in computer science from the University of Maryland, College Park, in 2001 and the M.S. degree in computer graphics and mobile robotics from the University of Washington, Seattle.

He is currently a Postdoctorial Researcher in the Department of Computer Science III, University of Bonn. His research interests include probabilistic sensor fusion and state estimation techniques with applications in mobile robotics and intelligent environments. He was a part of the University of Washington team that collaborated with researchers at SRI on the Centibots DARPA project from 2001 to 2002. Since 2003, he has gone on to build distributed web applications and real-time data warehouses. Currently, he works at Microsoft creating technologies to help better diagnosis and troubleshoot computer systems in the hopes of one day arriving at a "self-healing" computer.