SLAM in Large Indoor Environments with Low-Cost, Noisy, and Sparse Sonars

Teddy N. Yap, Jr. and Christian R. Shelton
Department of Computer Science and Engineering
University of California, Riverside, CA 92521, USA
{tyap, cshelton}@cs.ucr.edu

***Abstract*— Simultaneous localization and mapping (SLAM) is a well-studied problem in mobile robotics. However, the majority of the proposed techniques for SLAM rely on the use of accurate and dense measurements provided by laser rangefinders to correctly localize the robot and produce accurate and detailed maps of complex environments. Little work has been done on the use of low-cost but noisy and sparse sonar sensors for SLAM in large indoor environments involving large loops. In this paper, we present our approach to SLAM with sonar sensors by applying particle filtering and a line-segment-based map representation with an orthogonality assumption to map indoor environments much larger and more challenging than those previously considered with sonar sensors. Results from robotic experiments demonstrate that it is possible to produce good maps of large indoor environments with large loops despite the inherent limitations of sonar sensors.**

I. INTRODUCTION

Since it was first introduced by Smith and Cheeseman [1], the simultaneous localization and mapping (SLAM) problem has become one of the mainstream research areas in mobile robotics. The SLAM problem involves estimating the position and orientation of the robot while building a map of the environment in parallel. In order to accomplish SLAM, the robot is usually equipped with sensors (e.g. wheel encoders, range sensors, cameras) that allow it to observe (though only partially and inexactly) the state of the world including itself. The SLAM problem is inherently difficult and complex although the specific sensor used also contributes to the hardness of SLAM.

SLAM is a well-studied problem in mobile robotics and a number of techniques have been proposed. However, the majority of the proposed techniques for SLAM rely on the use of accurate and dense measurements provided by laser rangefinders to correctly localize the robot and produce accurate and detailed maps of complex environments (see e.g. [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]). Relatively little work has been done on the use of low-cost but noisy and sparse sonar sensors for SLAM in large indoor environments involving large loops. Obviously, the SLAM problem is much more difficult and challenging in the case of sonar sensors than laser rangefinders (which have become the *de facto* range sensors for SLAM). Despite the difficulty, we believe that it is interesting to investigate the extent to which sonar sensors can be used for SLAM especially in mapping large indoor environments with large loops.

Although sonar sensors are not as accurate and do not provide as dense observations as laser rangefinders, they are still an attractive alternative to laser rangefinders when it comes to cost, power consumption, size and weight, and computational requirements. Compared to laser rangefinders, sonars cost several orders of magnitude less, consume less power, are small and lightweight, and impose minimal computational requirements. As such, sonar sensors are well-suited for use in inexpensive consumer-oriented and minimally-equipped robots that are typically limited in both power and computational capability.

In this paper, we present our approach to SLAM with sonar sensors by applying particle filtering and an orthogonal line-segment-based map representation to map indoor environments much larger and more challenging than those previously considered with sonar sensors. Rather than employing a grid-based representation for the map, we use a feature-based representation where the features are represented as line segments. Line segments are suitable for compactly describing most structured indoor environments that are usually composed of walls, doors, glass windows, etc. that are either parallel or perpendicular to each other. Similar to [11], we also make use of the *orthogonality* assumption about the shape of the environment in order to reduce the complexity, mapping only lines that are either parallel or perpendicular to each other.

A major difficultly in using line segments as an environment's representation is extracting them from noisy and sparse sensors such as sonars (in our case, an array of 16 sonar transducers) compared to a single dense scan of a 180° laser rangefinder. Thus, in this work, we adopt the *multiscan* approach [12] to group consecutive sparse scans so that measurements from multiple time frames can be used to extract line segments, although our work differs on how the sparse scans are collected and the frequency at which the feature extraction is performed.

The contributions of this paper can be summarized as follows. We show through extensive experiments that it is possible to produce good quality maps of large indoor environments with large loops even with noisy and sparse sonar sensors. To our knowledge, the environments we consider in our experiments are much larger and more challenging than those previously reported in the literature for SLAM with sonars. The results provide significant supportive evidence for the potential viability of sonars for large-scale indoor SLAM. Our method employs a particle filtering technique where each particle carries a single map rather than a distribution over possible maps. Moreover, we apply the orthogonality assumption to reduce the complexity. Finally, we discuss how we extract line segments from sonar data and we introduce a simple sensor model for computing the likelihood of the observations.

II. PRIOR WORK

While much of the work on SLAM with proximity sensors has focused on laser scanners (see e.g. [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]), some researchers have used sonars. In this section, we briefly describe some of the techniques proposed for carrying out SLAM with sonars. There are two main criteria that can be used to categorize existing sonar-based SLAM techniques: the representation used to model the environment and the technique used to estimate the state belief distribution.

Zunino and Christensen [13] described an algorithm for SLAM based on the extended Kalman filter (EKF). The EKF approach was used to build and maintain the map of the environment. Their method used point features as landmarks (representing corners, edges, and thin poles detectable by standard sonar sensors through a triangulation technique). They also presented a method for detecting the failures of the EKF approach and recovering from such failures. Experiments were performed using a Nomadic SuperScout mobile robot in a living room of size $5\text{m} \times 9\text{m}$.

Tardós et al. [14] described a technique for the creation of feature-based stochastic maps using standard Polaroid sonar sensors. In their work, they used the Hough transform [15] for detecting point features (representing corners and edges) and line segment features (representing walls) from sonar data acquired from multiple uncertain vantage points. Instead of building one global map from the start, they generated a sequence of local maps of limited size, and then joined them together, to obtain the global map. Techniques for joining and combining several stochastic maps were presented. The locations of geometric features in the environment and the position of the robot were jointly estimated in a stochastic framework via the EKF. Experiments were carried out using a B21 mobile robot equipped with a ring of 24 Polaroid sensors in a $12\text{m} \times 12\text{m}$ environment.

A similar approach to that of Tardós was proposed by Leonard et al. [16] except that they incorporated past robot positions to the state vector and explicitly maintained the estimates of the correlations between current and previous robot states. By doing so, it became possible to consistently initialize new map features by combining data from multiple vantage points. Experiments were conducted in a testing tank of size $10\text{m} \times 3\text{m} \times 1\text{m}$ for underwater sonar-based SLAM, a simple “box” environment made of plywood, and along a 25m long corridor.

A much earlier work along the same lines of feature-based stochastic mapping using the EKF was given by Rencken [17] although experiments were only performed in a simulation of a $5\text{m} \times 5\text{m}$ room.

Instead of jointly estimating the robot state and the locations of line segment features via the EKF, Lorenzo et al. [18] used the EKF to estimate only the robot state. The environment was described by a global segment-based map that was built using local sonar-based occupation maps to identify obstacle boundaries. The Hough transform was used to extract the segments that represent the obstacle boundaries. The segments were then incorporated to the global segment-based map. The Hough-based approach provided a correction of the estimated robot pose which was integrated with odometric information via the EKF. The approach was tested with a Nomad 200 mobile robot equipped with a ring of 16 sonar sensors traversing a corridor environment.

More recently, Schröter, Böhme, and Gross [19] presented a combination of map-matching with a Rao-Blackwellized particle filter (RBPF) [20] which enabled them to solve the SLAM problem with low-resolution sonar range sensors. They introduced a simple and fast but very efficient shared representation of gridmaps which reduced the memory cost overhead caused by inherent redundancy between the particles. Experimental results were presented with a SCITOS A5 robot platform navigating in a home store environment.

Other notable recent related work on SLAM with sparse sensors include the work of Beevers and Huang [12] on SLAM with sparse sensing using the multiscan approach and RBPF, Abrate, Bona, and Indri [21] on experimental EKF-based SLAM for mini-rovers with IR sensors only, and Choi, Lee, and Oh [22] on a line based SLAM with infrared sensors using geometric constraints and active exploration.

The focus of our work reported in this paper is on SLAM in large structured indoor environments involving large loops using sonar sensors. The largest environment we consider has an approximate area of $70.0\text{m} \times 53.7\text{m}$ containing two major loops and is made up of various types of obstacles (e.g. brick walls, tiled walls, glass windows and doors, and cable railings). Our work differs from [19] in that we use a line-segment-based instead of a grid-based representation for the map. By adopting a line-segment-based map representation, we can avoid the usual problems associated with a grid-based representation such as data smearing [14], the strong independence assumption between the grid cells, and the considerable amount of memory required for storage. However, a major difficulty with line segments is extracting them from sparse and noisy sonars. To overcome the sparseness, we apply the multiscan approach [12] to group consecutive sparse scans. In order to reduce the complexity of SLAM, similar to [11], we make use of the orthogonality assumption about the shape of the environment by mapping only lines that are parallel or perpendicular to each other. Unlike [14], [16], [17], [18], we use particle filters to sample the distribution over the most recent robot poses. Each particle in our particle filter is associated with a single map rather than a distribution over possible maps.

III. OUR LOCALIZATION APPROACH

The basis of our approach is a particle filter, where each particle is an estimate of the recent m robot poses. Thus, the particles collectively sample the space of the recent m robot poses. Here we assume that the robot moves in a planar environment so that its pose at time t can be represented as $\mathbf{x}_t = (x_t, y_t, \theta_t)^T$, where $(x_t, y_t)^T$ is its Cartesian coordinates and θ_t is its heading or orientation with respect to some global reference frame. Each particle carries an estimated map of the environment that is represented as a set of line segments. Because of the orthogonality assumption about the shape of the environment, the map will contain only two types of lines: horizontal and vertical. Horizontal (vertical) lines are assigned to be parallel to the x-axis (y-axis) of the global reference frame. Extracted line segments that are not close to being either horizontal or vertical are simply discarded and not placed in the map.

A. Motion Model

Let the ith particle at time t be $\mathbf{p}_t^{[i]} = (\mathbf{x}_{t-m+1:t}^{[i]}, M_t^{[i]})$, where $\mathbf{x}_{t-m+1:t}^{[i]}$ is the sequence of the recent m robot poses $\mathbf{x}_{t-m+1}^{[i]}, \mathbf{x}_{t-m+2}^{[i]}, ..., \mathbf{x}_t^{[i]}$ according to particle i and $M_t^{[i]}$ is the set of line segments representing the map of particle i at time t. The particles move according to the probabilistic motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{c}_t)$, where $\mathbf{c}_t$ is the control command executed by the robot during the time interval $[t-1, t)$, to account for the inherent uncertainty in robot motion. The control command $\mathbf{c}_t$ is given by the pair $(d_t, r_t)^T$, where d_t is the distance traveled and r_t is the rotation made by the robot, according to the wheel encoders. Due to the probabilistic nature of the motion model, the particles spread and generate different possible robot trajectories. Here we use the motion model in [23]:

$$\begin{aligned} x_t &= x_{t-1} + \hat{d}_t \cos(\theta_{t-1} + \hat{r}_t) \\ y_t &= y_{t-1} + \hat{d}_t \sin(\theta_{t-1} + \hat{r}_t) \\ \theta_t &= (\theta_{t-1} + \hat{r}_t) \bmod 2\pi \ , \end{aligned}$$

where $\hat{d}_t$ and $\hat{r}_t$ denote the robot's true translation and rotation, respectively. The true translation and rotation both differ from the translation and rotation measured by the robot due to systematic and random errors. Specifically,

$$\begin{aligned} \hat{d}_t &= d_t + \delta_{\text{trans}}|d_t| + \epsilon_{\text{trans}} \\ \hat{r}_t &= r_t + \delta_{\text{rot}}|d_t| + \epsilon_{\text{rot}} \ , \end{aligned}$$

where δ_{trans} and δ_{rot} describe the systematic error and $\epsilon_{\text{trans}} \sim \mathcal{N}(0, \sigma_{\text{trans}}^2)$ and $\epsilon_{\text{rot}} \sim \mathcal{N}(0, \sigma_{\text{rot}}^2)$ are the additive random variables representing the random errors. In our experiments, $\delta_{\text{trans}} = 1.0 \times 10^{-2}$, $\delta_{\text{rot}} = 1.0 \times 10^{-5}$, $\sigma_{\text{trans}} = 2\text{mm}$, and $\sigma_{\text{rot}} = 2°$.

B. Sensor Model

When using particle filtering for state estimation, particles are usually assigned weights based on their current estimates and the current measurements or observations obtained about the system. The weights are assigned in such a way that likely (good) particles receive higher weights than unlikely (bad) particles. The weights are used during the resampling stage of the particle filter such that particles with higher weights are resampled with higher probability than particles with lower weights. Through resampling, it is hoped that good particles (those that can explain the data well) are retained while bad particles are eliminated. Typically, the sensor model (usually expressed in probabilistic form) is utilized in computing the appropriate weights for the particles. For SLAM, the weight assigned to particle i is commonly directly proportional to the likelihood of the observations $p(\mathbf{s}_t|\mathbf{x}_t^{[i]}, M_t^{[i]})$ where $\mathbf{s}_t = \{s_t^1, s_t^2, ..., s_t^K\}$ is the set of sensor measurements taken by the K available sensors at time t. From the preceeding discussion, it is evident that using an accurate and robust sensor model is therefore crucial to the success of the particle filter and other state estimation techniques in general. However, deriving an accurate and robust model for sensors is generally a difficult problem.

In this paper, we propose a heuristic but simple way to calculate the weights of the particles which is inspired by the work of Schröter, Böhme, and Gross [19] on map matching. In [19], they need to match a local map to a global one. They calculate the weight of particle i as

$$w_t^{[i]} = e^{\frac{m_t^{[i]}}{f}} \ , \qquad (1)$$

where f is a free parameter that influences the spread of the particle weights and $m_t^{[i]}$ is a measure of the quality of the match. [19] employs a grid-based map, so we replace their definition of $m_t^{[i]}$ with our own. For each sonar range measurement s_t^k obtained by sensor k at time t, we perform ray tracing along the facing direction of sensor k twice: a) using the current map $M_t^{[i]}$ and b) using the current map but with the line segments extended by a certain length of L_{ext} (e.g. 200mm) at both ends.[1] For each ray tracing operation for sensor k at time t, we calculate the "true" (expected) range measurement[2] s_t^{k*} for sensor k at time t. If the absolute difference between the actual and expected range measurements is less than or equal to the standard deviation of the sonar measurements σ_d (50mm) (i.e. $|s_t^k - s_t^{k*}| \le \sigma_d$), we count it as $+1$. Otherwise, we count it as -1. We then take the average of the counts obtained from the two ray tracing operations for sensor k and use that as the sensor's contribution. We sum these contributions to obtain $m_t^{[i]}$. Fig. 1 illustrates how the match value is computed. In this way, a particle whose map can explain the current measurements well will be assigned a higher weight than one that cannot. We should point out that the above procedure to compute the match value is just a simple heuristic that is found to work well in our experiments. We believe that better scheme for computing the particle weights would produce better particle filter performance.

[1]We exclude those measurements s_t^k that are equal to the maximum sonar range s_{max} (e.g. 5000mm) since they provide us with little information as to the location of objects or obstacles.

[2]The "true" (expected) range measurement s_t^{k*} is intended to be the distance to the nearest obstacle in the map along the facing direction of the sensor. If the ray doesn't intersect with any obstacle in the map, $s_t^{k*} = +\infty$.

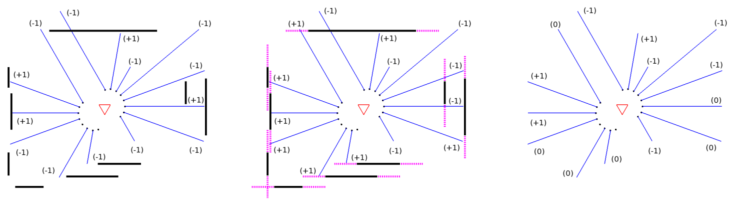

Fig. 1. Computation of the match value. (Left) Result of the first ray tracing operation using the estimated line segments. Count values are shown in parenthesis beside their corresponding sonar measurements. Missing sonar measurements are equal to $s_{\max}$ and are not considered in computing the match value. (Middle) Result of the second ray tracing operation using the extended line segments. (Right) Average count value for each sonar measurement. The match value $m_t = -2$ for this example.

C. Particle Filtering

It is well-known that the resampling stage of the particle filter can eliminate the correct particle. This is called the *particle depletion problem* [7]. In order to reduce the risk of particle depletion, we also adopt the selective resampling approach reported in [7] by computing the effective number of particles $N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N}(w^{[i]'})^2}$ (where $w^{[i]'} = \frac{w^{[i]}}{\sum_{i=1}^{N} w^{[i]}}$) and resampling only when $N_{\text{eff}} < N/2$, where N is the sample size of the particle filter.

The following summarizes our sonar SLAM approach:
For time $t = 1, 2, ..., T$:

1) For each particle $\mathbf{p}_{t-1}^{[i]}, i = 1, 2, ..., N$:
   a) Update the position of particle $\mathbf{p}_{t-1}^{[i]}$ by sampling from the motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{[i]}, \mathbf{c}_t)$ to get the new position $\mathbf{x}_t^{[i]}$ and thus $\mathbf{p}_t^{[i]}$.
   b) Update the weight of particle $\mathbf{p}_t^{[i]}$ according to: $w_t^{[i]} \leftarrow e^{\frac{m_t^{[i]}}{f}} \times w_{t-1}^{[i]}$
2) Compute the effective number of samples N_{eff}.
3) If $N_{\text{eff}} < N/2$, resample from the set of weighted particles $\{(\mathbf{p}_t^{[i]}, w_t^{[i]'})\}_{i=1}^{N}$ with replacement with each particle having the probabilty of being selected proportional to the normalized weight $w_t^{[i]'}$ and set weights $w_t^{[i]}$ to 1.
4) For each particle $\mathbf{p}_t^{[i]}, i = 1, 2, ..., N$:
   a) Extract line segment features from recent m sonar scans $\mathbf{s}_{t-m+1:t}$ and recent m poses $\mathbf{x}_{t-m+1:t}^{[i]}$.
   b) Incorporate extracted line segments to map $M_t^{[i]}$.
   c) Adjust map $M_t^{[i]}$.

In the next section, we discuss how we extract line segment features from the group of recent m (e.g. 15) sonar scans $\mathbf{s}_{t-m+1:t}$ and how the line segments are incorporated into the map as well as how the map is maintained.

IV. OUR MAP BUILDING APPROACH

A. Line Segment Extraction

For extracting line segments from the group of recent m sonar scans, we use the randomized Hough transform (RHT) [24] technique proposed by Kultanen, Xu, and Oja. Before line segments can be extracted, we first plot each sonar measurement as a point (in the global reference frame) that represents the nominal position of the sonar return, computed along the central axis (facing direction) of the transducer, with respect to the robot pose where the measurement was taken. Let P be the set of such points. We then use the RHT to find groups of (almost) collinear points in P and extract the parameters of the lines that fit those groups of points.

The Hough transform [15] is a technique used in digital image processing for extracting features such as lines and curves in binary edge images. It is a voting scheme where each point (pixel) in the image votes for a set of features (lines, curves) that pass through it. Voting is performed in a discretized parametric space, called the *Hough space*, representing all possible feature locations. The most voted cells in the Hough space should correspond to the features actually present in the image.

The RHT is an improvement on the original Hough transform by reducing the computation time and memory usage. The basis of the method lies on the fact that a single parameter space point can be determined uniquely with a pair, triple, or generally n-tuple of points from the image. Such an n-tuple of points can be chosen randomly from the image and hence the name. Unlike in the original Hough transform where each point in the image votes for a set of cells in the Hough space, a group of n randomly chosen points from the image votes for only one parameter space point in the RHT. The presence of a specific feature in the image is quickly revealed by the accumulation of a small number of parameter space points. Rather than having a fixed-size array structure for implementing the Hough space (also called the *accumulator space*) as in the original Hough transform (thus imposing some predefined accuracy in parameter point location), in RHT, it is possible to use a dynamic tree structure to store the accumulator cells with non-zero votes in the parameter space. This way, one can achieve as high an accuracy as required while at the same time bringing memory usage to near minimal.

To use the RHT for line segment extraction, we use the polar coordinate parametrization (ρ, θ) for lines, where ρ is the length of the normal from the origin to the line and θ is the angle that the normal makes with the positive x-axis. Using this parametrization, the equation of the line can be written as

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \ . \quad (2)$$

To ensure unique parametrization of lines, we impose that $0 \le \rho \le \rho_{max}$ and $-180° \le \theta < 180°$.

In our implementation, we discretize the Hough space using the resolutions $\Delta\rho = 50$mm and $\Delta\theta = 1°$. Let A be the fixed-size accumulator array for implementing the discretized Hough space. A point (ρ, θ) in the Hough space corresponds to the accumulator cell A_{ij} with $i = \lfloor \rho/\Delta\rho \rfloor$ and $j = \lfloor (\theta - \theta_{min})/\Delta\theta \rfloor$. The following outlines the RHT to extract line segments from the set P:

While P contains at least N_0 points and the maximum number of trials has not been reached:

1) Reset the accumulator cells A_{ij} to 0.
2) While the accumulator array A does not have a global maximum that exceeds a threshold τ (e.g. 100)
   a) Pick two points p_a and p_b randomly from P.
   b) Solve the line parameters (ρ, θ) from the line equation with points p_a and p_b.
   c) Increment the accumulator cell A_{ij} corresponding to (ρ, θ) by 1.
3) Let $(\hat{\rho}, \hat{\theta})$ be the line determined by the location of the maximum in A.
4) Let Q be the set of points in P that are close to the line $(\hat{\rho}, \hat{\theta})$.
5) If Q contains at least N_0 points, use Q and $(\hat{\rho}, \hat{\theta})$ to extract line segments.
6) Remove from P points in Q used to generate segments.

Prior to extracting line segments in Step 5 above, we apply the orthogonality assumption and proceed with the line segment extraction only if the line $(\hat{\rho}, \hat{\theta})$ is close to being horizontal or vertical. This is done by testing whether $\hat{\theta}$ is within a certain threshold $\epsilon_\theta = 5°$ from $90°$ or $-90°$ for horizontal line and $-180°$, $0°$, or $180°$ for vertical line. If so, we then set $\hat{\theta}$ to be one of $\{-180°, -90°, 0°, 90°\}$ accordingly and update $\hat{\rho}$ so as to best fit the points in Q in the least-square sense. We denote the line resulting from applying the orthogonality assumption as $(\hat{\rho}', \hat{\theta}')$.

Given $(\hat{\rho}', \hat{\theta}')$, we project the points in Q onto the line $(\hat{\rho}', \hat{\theta}')$. Let $R = \{p'_1, p'_2, ...\}$ be the set of projected points arranged sequentially starting from one of the extreme endpoints. We then sequentially partition R into subsets $S_h, h = 1, 2, ...,$ with each S_h representing the set of points belonging to a line segment. We break the line into segments if there is a gap greather than a threshold $L_{\text{sep}} = 500$mm. After partitioning R into subsets $S_h, h = 1, 2, ...,$ the line segments are easily obtained as those that connect the extreme endpoints in each S_h. To increase the reliability of the line segment extraction phase, only those line segments that are made up of at least $N_0 = 8$ points and with length at least $L_{\text{min}} = 200$mm are incorporated into the map.

B. Line Merging and Map Management

When incorporating a line segment into the map, we first test whether it can be merged with the line segments of the same type already in the map. Two line segments can be merged if

- they overlap and the perpendicular distance between them is less than or equal to L_{dist} (300mm); or

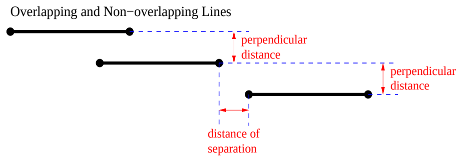

Fig. 2. The perpendicular distance and distance of separation between overlapping and non-overlapping horizontal lines.

- they do not overlap but the perpendicular distance between them is less than or equal to L_{dist} and their distance of separation is less than or equal to L_{sep}.

Fig. 2 illustrates the perpendicular distance and distance of separation between overlapping and non-overlapping horizontal lines. A similar interpretation exists for vertical lines.

To merge two line segments, we first compute the position (i.e. the ρ-value) of the resulting line segment. The position of the resulting line segment is the sum of the positions of the given line segments, weighted by their respective number of points. We then project the endpoints of the given line segments onto the resulting line and the two projections that are farthest apart define the endpoints of the resulting line segment. The number of points of the resulting line segment is just the sum of the number of points of the given line segments. If a line segment cannot be merged with the line segments already in the map, it is simply added to the map.

After incorporating the line segments into the map, we continue to merge segments that meet our criteria until no further mergings are possible. This step can have the desirable effect of integrating two distinct line segments representing the same environmental feature into one (e.g. a long stretch of wall occassionally occluded by dynamic obstacles or relatively small and insignificant objects).

V. EXPERIMENTAL RESULTS

We have implemented our SLAM approach for sonars and tested it on the ActivMedia robotics P3-DX mobile robot platform. The robot is equipped with a front sonar array with eight sensors, one on each side and six facing forward at $20°$ intervals. It also has a rear sonar array with eight sensors, one on each side and six facing backward at $20°$ intervals. Therefore, a single sonar scan yields a total of 16 sonar measurements. Of the 16 sonar measurements in a single scan, only a small fraction actually corresponds to correct measurements while the rest are unreliable due to angular uncertainty ($20°$ to $30°$ for sonars), specular and multiple reflections, crosstalk, etc. This is in stark contrast to typical laser rangefinders that produce accurate (with a statistical error of $\approx$ 5mm in range measurement and angular beam width of $\le 1°$ for each beam) and dense measurements (e.g. 180 or 360 measurements in each scan) and are not affected by problems such as specular reflections and crosstalk. As such, SLAM with sonars is much more difficult and challenging than SLAM with laser rangefinders. Despite the shortcomings of sonars, our experimental results show that it is possible to produce good quality maps of large indoor environments with large loops using sonars.

In our experiments, the robot is controlled by an IBM ThinkPad X32 notebook computer and navigated around different environments by visiting predefined waypoints while collecting control and sensor data along the way. We considered three test environments of increasing sizes and complexities. The first test environment is a makeshift environment that represents a scaled-down version of a typical office environment. The associated map of the environment has an approximate size of $6.7\text{m} \times 6.7\text{m}$. It is mostly made up of smooth cardboard walls. The second test environment is a portion of the faculty suite on the third floor of our Engineering Building II. It is bigger than the first with an approximate map size of $38.6\text{m} \times 12.7\text{m}$ and contains two major loops. Finally, the third test environment is the entire hallway of the third floor of our Engineering Building II, including the two bridges connecting to the old engineering building. It is the biggest environment in our experiments with a map size of approximately $70.0\text{m} \times 53.7\text{m}$ and contains two major loops. The third test environment is made up of various types of obstacles such as brick walls, tiled walls, cable railing, glass windows, trash bins, etc. Unlike the first two test environments, the third test environment is dynamic with people walking along the corridors during the experiments. Fig. 3 (cols. 1 and 2) shows the three test environments and their associated maps while Table I provides summary information about our experiments.

To show the effectiveness of our SLAM approach using sonars to compensate for odometric errors, we show in Fig. 3 (col. 2) the trajectory from the raw encoder readings (broken lines) against the desired path of the robot (solid lines) for all three test environments. It is evident from Fig. 3 (col. 2) that the robot's odometry suffers from drift that gets more pronounced in larger environments. Therefore, relying only on the robot's odometry for performing SLAM is not sufficient and sonar measurements taken must be used for correcting the robot's pose. Fig. 3 (col. 3) shows the resulting maps and robot trajectories with our approach for the three test environments. Although the generated maps are not exactly the same as the true maps, they do capture the main structure of the environments. Additionally, our approach managed to close the loops properly in all test environments which is generally considered a difficult problem in SLAM. Lines in our maps correspond to actual major obstacles such as walls, glass windows, doors, cable railings, as well as to some minor ones such as trash bins and posts. Because of the Hough transform, our line extraction procedure is quite robust to noise caused by specular and multiple reflections and phantom readings. Also, since we only consider horizontal and vertical lines, our line extraction procedure is effective at filtering out dynamic objects particularly for the third test environment.

We also performed experiments without using the orthogonality assumption and the final maps and robot trajectories are shown in Fig. 3 (col. 4). Without the orthogonality assumption, the resulting maps and robot trajectories are not properly estimated and corrected. Finally, we also implemented a RBPF using gridmaps for SLAM based on map matching [19] and the resulting maps for all three test environments are also shown in Fig. 3 (col. 5). Note that we used the same set of parameter values for generating the results for all three test environments with our approach (except for the number of particles N) while we had to use different sets of parameter values to generate the resulting gridmaps shown in Fig. 3 (col. 5) using RBPF with map matching. We can easily see the effect of data smearing when using a cell-based approach to SLAM; measurements are often blurred onto a region of the map to account for angular and distance uncertainty.

VI. CONCLUSIONS

SLAM has received considerable attention in the mobile robotics community for the last two decades. However, much of the research effort for SLAM has focused on the use of highly accurate and dense measurements provided by laser rangefinders to correctly localize the robot and produce accurate and detailed maps of complex environments. In this paper, we presented an approach to SLAM for a mobile robot equipped with low-cost but noisy and sparse sonar sensors navigating in large indoor environments involving large loops. The proposed approach applies particle filtering where each particle is an estimate of the recent robot poses and carries a map of the environment represented as a set of line segments. To overcome the sparseness of sonars and to allow for the reliable extraction of line segment features from the environment, we used the multiscan approach and grouped consecutive sparse scans into multiscans. To reduce the complexity of SLAM particularly when sonars are used, we applied the orthogonality assumption about the shape of the environment by mapping only lines that are parallel or perpendicular to each other. The orthogonality assumption is reasonable especially for most man-made indoor environments where major structures such as walls, windows, and doors are either parallel or perpendicular to each other. The randomized Hough transform was used to extract line segments from multiscans and was quite robust to noise caused by specular and multiple reflections and phantom readings, problems usually associated with sonars. Despite the inherent limitations of sonars, results of empirical validation, carried out using a real mobile robot platform navigating in different environments of increasing sizes and complexities, provide supportive evidence for the potential viability of sonars for complex large-scale indoor SLAM.

REFERENCES

[1] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *IJRR*, vol. 5, no. 4, pp. 56–68, 1986.
[2] E. Brunskill and N. Roy, "SLAM using incremental probabilistic PCA and dimensionality reduction," in *Proc. IEEE ICRA'05*, Apr. 2005, pp. 342–347.
[3] A. Eliazar and R. Parr, "DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks," in *Proc. 18th IJCAI'03*, Aug. 2003, pp. 1135–1142.
[4] A. I. Eliazar and R. Parr, "DP-SLAM 2.0," in *Proc. IEEE ICRA'04*, Apr. – May 2004, pp. 1314–1320.
[5] ——, "Hierarchical linear/constant time SLAM using particle filters for dense maps," in *NIPS 18*, 2006, pp. 339–346.

TABLE I

EXPERIMENTS SUMMARY

	Test Environment 1	Test Environment 2	Test Environment 3
Map Size	$\approx 6.7\text{m} \times 6.7\text{m}$	$\approx 38.6\text{m} \times 12.7\text{m}$	$\approx 70.0\text{m} \times 53.7\text{m}$
Data Collection Time	$\approx$ 5 minutes	$\approx$ 9 minutes	$\approx$ 30 minutes
Distance Traveled	26.7m	85.7m	283.5m
Total Time Steps T	545	727	1,985
Number of Sonar Measurements (excluding $s_{\max}$)	5,915	7,577	19,348
Average Number of Sonar Measurements per Time Step (excluding $s_{\max}$)	10.9	10.4	9.7
Number of Particles Used N	200	300	200

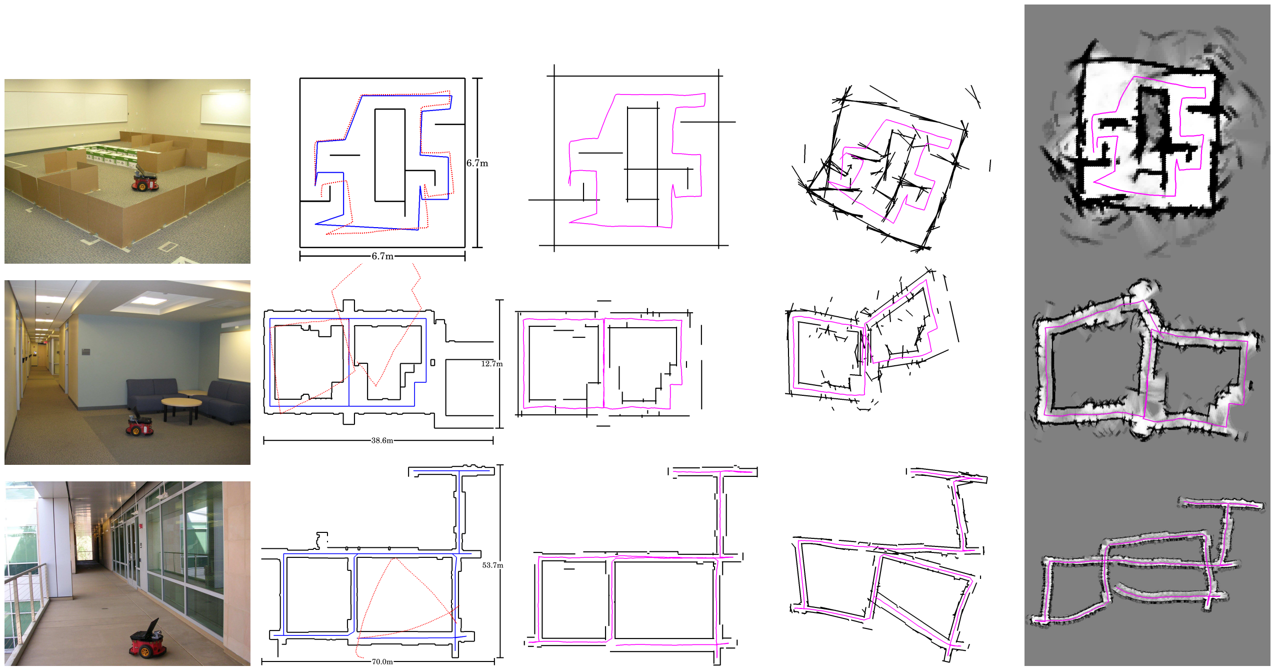

Fig. 3. The three test environments (col. 1) and their associated maps (col. 2). Col. 2 also shows the paths of the robot (solid lines) and the trajectory estimates based from encoder readings (broken lines). The estimated maps and robot trajectories with our approach (col. 3). Maps and robot trajectories estimated without using the orthogonality assumption (col. 4). Maps and robot trajectories estimated using a RBPF using grid maps (col. 5). For the grid maps, dark regions indicate high level of occupancy, white regions indicate low level of occupancy, and gray regions represent unexplored areas.

[6] A. Garulli, A. Giannitrapani, A. Rossi, and A. Vicino, "Mobile robot SLAM for line-based environment representation," in *Proc. 44th IEEE CDC-ECC'05*, Dec. 2005, pp. 2041–2046.

[7] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with Rao-Blackwellised particle filters by adaptive proposals and selective resampling," in *Proc. IEEE ICRA'05*, Apr. 2005, pp. 2432–2437.

[8] 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 IROS'03*, Oct. 2003, pp. 206–211.

[9] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. 18th AAAI'02*, July – Aug. 2002, pp. 593–598.

[10] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *Proc. IEEE ICRA'03*, vol. 2, Sept. 2003, pp. 1985–1991.

[11] V. Nguyen, A. Harati, A. Martinelli, R. Siegwart, and N. Tomatis, "Orthogonal SLAM: a step toward lightweight indoor autonomous navigation," in *Proc. IEEE/RSJ IROS'06*, Oct. 2006.

[12] K. R. Beevers and W. H. Huang, "SLAM with sparse sensing," in *Proc. IEEE ICRA'06*, May 2006, pp. 2285–2290.

[13] G. Zunino and H. I. Christensen, "Navigation in realistic environments," in *Proc. 9th SIRS'01*, July 2001, pp. 405–413.

[14] J. D. Tardós, J. Neira, P. M. Newman, and J. J. Leonard, "Robust mapping and localization in indoor environments using sonar data," *IJRR*, vol. 21, no. 4, pp. 311–330, Apr. 2002.

[15] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Comm. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.

[16] J. J. Leonard, R. J. Rikoski, P. M. Newman, and M. Bosse, "Mapping partially observable features from multiple uncertain vantage points," *IJRR*, vol. 21, no. 10–11, pp. 943–975, Oct. – Nov. 2002.

[17] W. D. Rencken, "Concurrent localisation and map building for mobile robots using ultrasonic sensors," in *Proc. IEEE/RSJ IROS'93*, July 1993, pp. 2192–2197.

[18] J. M. P. Lorenzo, R. Vázquez-Martín, P. Núñez, E. J. Pérez, and F. Sandoval, "A Hough-based method for concurrent mapping and localization in indoor environments," in *Proc. IEEE RAM'04*, Dec. 2004, pp. 840–845.

[19] C. Schröter, H.-J. Böhme, and H.-M. Gross, "Memory-efficient gridmaps in Rao-Blackwellized particle filters for SLAM using sonar range sensors," in *Proc. 3rd ECMR'07*, Sept. 2007.

[20] K. P. Murphy, "Bayesian map learning in dynamic environments," in *NIPS 12*, 2000, pp. 1015–1021.

[21] F. Abrate, B. Bona, and M. Indri, "Experimental EKF-based SLAM for mini-rovers with IR sensors only," in *Proc. 3rd ECMR'07*, Sept. 2007.

[22] Y.-H. Choi, T.-K. Lee, and S.-Y. Oh, "A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot," *Auton. Robot.*, vol. 24, no. 1, pp. 13–27, 2008.

[23] N. Roy and S. Thrun, "Online self-calibration for mobile robots," in *Proc. IEEE ICRA'99*, May 1999, pp. 2292–2297.

[24] P. Kultanen, L. Xu, and E. Oja, "Randomized Hough transform (RHT)," in *Proc. 10th ICPR'90*, vol. 1, June 1990, pp. 631–635.