Mobile Robot Localization: Integrating Measurements from a Time-of-Flight Laser

Ulf Larsson, Johan Forsberg, and Åke Wernersson, *Member, IEEE*

***Abstract*— This paper presents an algorithm for environment mapping by integrating scans from a time-of-flight laser and odometer readings from a mobile robot. The range weighted Hough transform (RWHT) is used as a robust method to extract lines from the range data. The resulting peaks in the RWHT are used as feature coordinates when these lines/walls are used as landmarks during navigation. The associations between observations over the time sequence are made in a systematic way using a decision directed classifier. Natural geometrical landmarks are described in the robot frame together with a covariance matrix representing the spatial uncertainty. The map is thus built up incrementally as the robot moves. If the map is given in advance, the robot can find its location and navigate relative to this *a priori* given map.**

Experimental results are presented for a mobile robot with a scanning range measuring laser having 2-cm resolution. The algorithm was also used for an autonomous plastering robot on a construction site. The sensor fusion algorithm makes few erroneous associations.

I. Introduction

This paper presents a systematic method for integrating scans from a time-of-flight laser on board a mobile robot. Experimental tests for indoor navigation have been completed. The method can be extended for outdoor navigation.

Previous work in this area has used either a grid based approach (i.e., uncertainty grids [6]) or a symbolic approach where features are extracted and interpreted [2]–[14]. The current work uses the symbolic approach with the map stored as a set of feature coordinates relative to the robot and a corresponding covariance matrix.

There are two basic types of uncertainty. First, the model of the world and the robot's motion are uncertain and the range measurements are noisy and contain spurious large errors. Secondly, the *identity* of the observed features are *unknown* giving rise to scene interpretation and association problems. The algorithm developed deals with both these uncertainties in a systematic way. The robustified estimation is performed using a Kalman filter based approach while the association uses an approximate Bayesian classifier.

The sensor fusion algorithms are robust enough to handle both the above mentioned spuriousness in the range measurements and large outliers in the odometer measurements, as when the wheels slip on some object on the floor. The algorithms can be extended to include a vision system—theoretical and experimental work is in progress.

Manuscript received February 9, 1995; revised November 30, 1995.

The authors are with the Department of Robotics & Automation, Luleå University of Technology, S-971 87 Luleå, Sweden.

Publisher Item Identifier S 0278-0046(96)02376-3.

A. Outline of the Measurement Integration Algorithm

The map generation process integrates range scan using the following four steps:

1) Take a rangescan and extract the feature parameters using the range weighted Hough transform and a robust least squares algorithm.
2) Predict the map state at the time when the scan was taken, with the motion estimated from the odometers. (The dead reckoning is a continuous process operating in parallel with the other calculations.)
3) Match the observations from the last scan with the predicted estimates in the map. Decide which observations are new features previously not present in the map. Ambiguous measurements are discarded.
4) Update the map using the matched observations using an extended Kalman filter. Extend the map with any new features.

In step 3 we define the association function $A_i()$ which is an approximate estimate of the probability that a certain matching is correct. The match will be accepted only if $A_i() > T$, otherwise the observation is discarded. In step 4, the actual estimation of the state vector from the observations and odometer measurements are performed using an extended Kalman filter [1].

II. Representation of a Two Dimensional Indoor Environment in Robot Coordinates

The map generation algorithm presented is intended for indoor navigation using a scanning laser to measure ranges to objects in the horizontal plane. The most prominent features of an indoor scene are the straight lines corresponding to the walls. These can be detected efficiently using the algorithms described in Section III.

A. Feature Representation

In this paper the *world* is modeled as a set of infinitely long *straight lines* (walls); we use the term feature to refer to the element in the environment causing the line in the range scan. Each feature is described by it's *orthogonal distance* (d) and *direction* (γ) from the robot (see Fig. 2). These two parameters are referred to as the feature parameters or feature coordinates. We write the feature parameter for feature i as

$$X_i(k) = \begin{bmatrix} d_i(k) \\ \gamma_i(k) \end{bmatrix} \tag{1}$$

Assuming that the modeled part of the world contains some finite number N of such features, then at time t_k the features in the robots environment are described by the state vector

$$X(k) = [d_1(k)\ \gamma_1(k)\ d_2(k)\ \gamma_2(k) \cdots d_N(k)\ \gamma_N(k)]^T.$$

Because features are described relative to the robot it follows that there is no fixed coordinate system and no explicit representation of the robot's position. The state vector could also contain other features like "jump edges" at range discontinuities, corners or, for outdoor navigation, tree trunks. The algorithm given below can be modified to included these cases.

B. The Motion Equations

The displacement of the robot from time t_{k-1} to t_k is described by the vector

$$U(k) = [p_x(k)\ p_y(k)\ p_\theta(k)]^T$$

given in the *robot frame* at time t_{k-1}. The first two components (p_x, p_y) are the translation and the last component (p_θ) is the rotation. In the robot used during the tests, the actual displacement is estimated from odometric data using the dead reckoning equations given in [11]. The state transition of the N feature coordinates are

$$\begin{aligned} X(k) &= f(X(k-1),\ U(k)) \\ &= X(k-1) + G(X(k-1))U(k) \end{aligned} \tag{2}$$

where

$$G(X(k-1)) = \begin{bmatrix} -\cos(\gamma_1(k-1)) & -\sin(\gamma_1(k-1)) & 0 \\ 0 & 0 & -1 \\ -\cos(\gamma_2(k-1)) & -\sin(\gamma_2(k-1)) & 0 \\ 0 & 0 & -1 \\ \cdots & \cdots & \cdots \\ -\cos(\gamma_N(k-1)) & -\sin(\gamma_N(k-1)) & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$

This equation is a kinematic model describing how the features positions change as the robot moves. Note that the state transition is *linear* in the motion $U(k)$ but not in the state vector $X(k)$. Observe that it is also possible to use this type of formalism for isolated objects such as tree trunks etc.

C. Two Transformations to Fixed Coordinate Systems

We have described how feature are described relative to the robot. It is also possible to represent the robot's position, or the position of an object, relative to two or more features. Two examples of this are given below.

By introducing a local coordinate system attached to a pair of features with parameters (d_1, γ_1) and (d_2, γ_2) relative to the robot, for example a corner in a room, it is possible to record the robot's absolute position or to reduce the size of the estimated state vector. The intersection of these two lines defines the origin of an ON-coordinate system. Choosing the

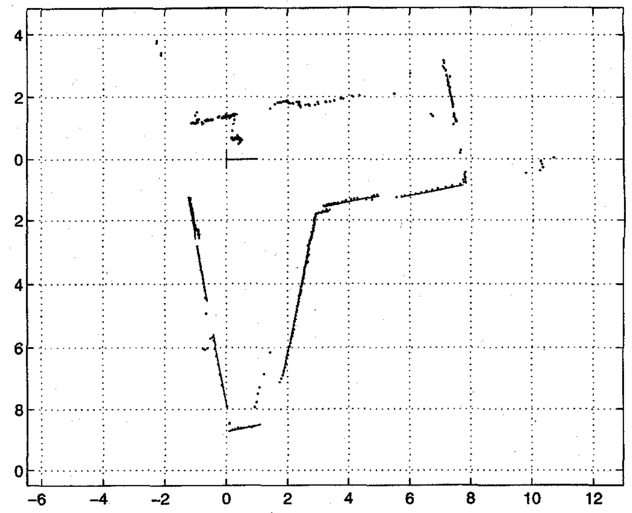

Fig. 1. Range scan and map of a room. The range scan (shown as dots) consists of 450 range measurements over a 270° sector. The scale is in meters. The T-shaped symbol is the robot and the straight lines shows the walls detected in this range scan.

x-axis to be rotated $(\gamma_1(k) + \gamma_2(k))/2$ relative the robot frame gives the robot's position in this coordinate system as

$$\begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} = \begin{bmatrix} -(d_1(k) + d_2(k)) \cos\left(\frac{\gamma_2(k) - \gamma_1(k)}{2}\right) \\ (d_1(k) - d_2(k)) \sin\left(\frac{\gamma_2(k) - \gamma_1(k)}{2}\right) \\ -\frac{\gamma_1(k) + \gamma_2(k)}{2} \end{bmatrix}.$$

Other pairs of (d_i, γ_i) in the state vector can be transformed into the new fixed coordinate system.

A second coordinate system is defined by the robot position at time t_0. Rewriting (2) and solving for the position using a least square method gives

$$\begin{aligned} \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} &= [G(X(0))^T G(X(0))]^{-1} \\ &\quad \times G(X(0))^T [X(k) - X(0)]. \end{aligned} \tag{3}$$

III. Extracting Features From Range Measurements

A time-of-flight laser on the robot takes a horizontal range scan of the surroundings. The parameters of the walls are extracted from these measurements using the Range Weighted Hough Transform (RWHT) [7]. With this method walls can be found even in very cluttered rooms. An example of a range scan and the detected walls are shown in Fig. 1.

A. The Measurement Model

Consider a feature/wall with feature coordinates (d, γ). The laser pointing in direction φ_m then measures the range r_m modeled as

$$r_m = \varepsilon_m \frac{d}{\cos(\varphi_m - \gamma)} + (1 - \varepsilon_m) R(\varphi_m) + \nu_m \tag{4}$$

where ν_m is the random pulse to pulse range fluctuation. If the laser beam is reflected off the wall then the binary random

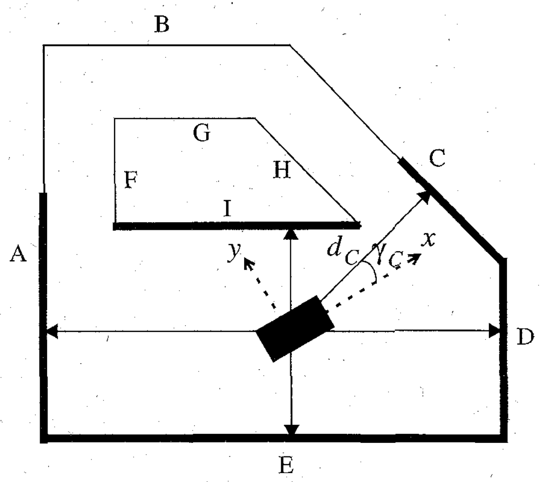

Fig. 2. Walls are described in the robot frame (the dashed arrows indicate the x- and y-axes) using the orthogonal distance d and the angle γ. The distance d is positive when the wall is facing the robot (A, B, C, D, and I), and negative when it is facing away from the robot (F, G, and H). The feature parameters d_C and γ_C are marked in the figure.

variable $\varepsilon_m = 1$. If, on the other hand, the laser beam is reflected off some other object at range $R(\varphi_m)$ then $\varepsilon_m = 0$. To estimate the parameters of the feature we need a method for extracting the set $\{r_m, \varphi_m \mid (\varepsilon_m = 1)\}$ from the range scan.

B. Extracting Feature Parameters with the Hough Transform

To extract the features parameters from the range scan we have to decide which measurements originate from features and which are from disturbing objects. From (4) we note that in the case of $\varepsilon_m = 1$ we get

$$r_m \cos(\varphi_m - \gamma) - d = \nu_m \cos(\varphi_m - \gamma).$$

It follows that most of the range measurements (r_m, φ_m) where $\varepsilon_m = 1$ satisfy

$$|d - r_m \cos(\varphi_m - \gamma)| < h \quad \text{where } h = 3\sqrt{\text{Var}(\nu_m)}. \quad (5)$$

The condition given in (5) is tested for a set of (d, γ) candidates. If the condition is satisfied for a candidate then the measurement votes for that candidate. This is the Hough transform (HT). When using range measurements a range weighted version of the HT is needed. Thus the range weighted Hough transform (RWHT) $C(d, \gamma)$ is given by

$$C(d, \gamma) = \sum_m w(r_m \cos(\varphi_m - \gamma) - d) r_m \quad (6)$$

where $w()$ is a window function. The argument in w is equal to the shortest distance between the point (r_i, φ_i) and the line (d, γ). Currently a unit rectangular window function $w(x)$ of width $2h$ is used

$$w(x) = \begin{cases} 1 & |x| \le h \\ 0 & |x| > h \end{cases}. \quad (7)$$

This choice allows an efficient implementation. Weighting with the range r_m is used since in the range scan the sampling is uniform with angular steps. Thus, as the distances increase, each sample corresponds to a longer surface segment, roughly proportional in length to the range r_m. For more details see [7] and [8].

The peaks in the RWHT are found by first searching for the single highest peak. The Hough peak is then enhanced using a robust least squares method giving the observation

$$Z_l = [d_l \quad \gamma_l]^T = X_i(k) + w_l(k)$$

where the identity between l and i is unknown. The measurements that fall within an interval

$$|d_l - r_m \cos(\varphi_m - \gamma_l)| < 2h$$

around this peak are then removed from the Hough transform and the procedure is repeated until all major peaks have been found. The RWHT should thus be seen as generating cluster gates. The resolution is not increased by making finer grids.

IV. Association and Estimation

A critical point in the position estimation process is the association of a measurement to a specific feature in the state vector. Since the individual measurements have no identity, we need an algorithm to decide which feature the measurement originates from. In pattern recognition this is know as unsupervised classification (see, for example, [5]). Because we want to build a map of the environment while the robot moves, the algorithm has to be recursive.

Association algorithms often use some kind of gate around the predicted value, either of a fixed size, or dependent upon the covariance, for example the Mahalanobis distance, [12]. This implies that the most probable match will always be chosen. Our algorithm refines this technique and uses the *best match only if the probability is high enough*, otherwise the measurement is discarded.

A. The State and Observation Vectors

From a range scan at time t_k the range weighted Hough transform gives a total of $L(k)$ features. Each extracted feature coordinate (d, γ) is denoted by $Z_l(k)$ where $0 \le l \le L(k)$. The corresponding covariance matrix is denoted by $S_l(k)$.

The total number of different landmarks *observed* up to and including time t_k is denoted $M(k)$. The state vector estimate $\hat{X}(k)$ in the Kalman filter at time t_k consists of these $2M(k)$ feature parameters as introduced in (1)

$$\hat{X}(k) = \begin{bmatrix} \hat{X}_1(k) \\ \hat{X}_2(k) \\ \cdots \\ \hat{X}_{M(k)}(k) \end{bmatrix}.$$

The estimated uncertainty covariance matrix of the estimate is denoted $\Sigma(k)$. The size of $\hat{X}(k)$ increases as the robot discovers new features, but might also shrink if features are discarded.

The state vector estimate can be initiated with an *a priori* map. In the case of no prior information the first set of peaks extracted at time t_0

$$\hat{X}(0) = \left[Z_1^T(0)\ Z_2^T(0) \cdots Z_{L(0)}^T(0)\right]^T$$
$$\Sigma(0) = \text{diag}(S_1(0),\ S_2(0), \ldots, S_{L(0)}(0))$$

gives the initialization.

B. The Basic Association Algorithm

We now return to the question of associating feature measurements $Z_l(k)$ to the features accumulated in the estimated state vector $\hat{X}(k)$. In particular, this subsection gives an algorithm for associating a single observation with the estimated state vector. From Section III-B it follows that we can write an observation $Z_l(k)$ of a feature as

$$Z_l(k) = [\eta_1 I \ \eta_2 I \cdots \eta_N I] X(k) + w_l(k) \tag{8}$$

where $\eta_1, \eta_2, \ldots, \eta_N$ are *binary* random variables with

$$\sum_{i=1}^{N} \eta_i = 1.$$

Based on all the previous observations, we wish to calculate the probability $\mathcal{P}_i$ that $Z_l(k)$ is an observation of feature i, $i \leq M(k)$.

$$\mathcal{P}_i = \Pr(\eta_i = 1 \mid \hat{U}(1), \ldots, \hat{U}(k), Z(1), \ldots, Z(k-1), Z_l(k)).$$

The previous information up to $\hat{U}(k)$ and $Z(k-1)$ is approximated by the state estimate and error covariance, giving the approximate probability

$$\mathcal{P}_i \approx \Pr(\eta_i = 1 \mid \hat{X}(k \mid k-1), \Sigma(k \mid k-1), Z_l(k), S_l(k)). \tag{9}$$

If Z_l is an observation of feature coordinates X_i, then the prediction of Z_l is $H_i \hat{X}(k \mid k-1)$, where

$$H_i = [I_1 \ I_2 \cdots I_{M(k)}] \qquad I_j = \begin{cases} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} & j = i \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} & j \neq i \end{cases}.$$

Using the corresponding covariance matrix, we get the likelihood function for Z_l being an observation of X_i as

$$P_i(Z_l, S_l) = \frac{e^{-\frac{1}{2}(Z_l - H_i\hat{X})^T (H_i \Sigma H_i^T + S_l)^{-1} (Z_l - H_i \hat{X})}}{2\pi\sqrt{|H_i \Sigma H_i^T + S_l|}} \tag{10}$$

where S_l is the covariance of the measurement Z_l. The time index is omitted for brevity.

Using Bayes rule, an *approximation* of the probability (7) for $1 \leq i \leq M(k)$ is given by the association function $A_i()$ defined as

$$\mathcal{P}_i \approx A_i(Z_l, S_l) = \frac{P_i(Z_l, S_l)}{\sum_{j=1}^{M} P_j(Z_l, S_l) + q} \tag{11}$$

where q models the likelihood that the measurement corresponds to a feature that is not yet in the estimated state vector ($M(k) < i \leq N$). This likelihood q can be thought of as the 'intensity of detectable objects' in the scene. Including the parameter q has a very significant effect on the behavior of the association algorithm and the shape of the decision regions, (see Fig. 3.)

The function $A_i(Z_l, S_l)$ gives us an estimate of the probability that Z_l is an observations of the feature i. In landmark based navigation it is not always desirable to use the best match as it is often very confusing to make even a single mismatch but quite harmless to just skip an observation. Thus only observations with $A_i() > T$ are accepted as observations of the feature parameters X_i. The threshold T is "close" to one and usually used as an tuning parameter. An illustration of the algorithms behavior is given in Fig. 3. When a sufficiently probable association has been found and accepted, the measurement is used to update the state using Kalman filtering.

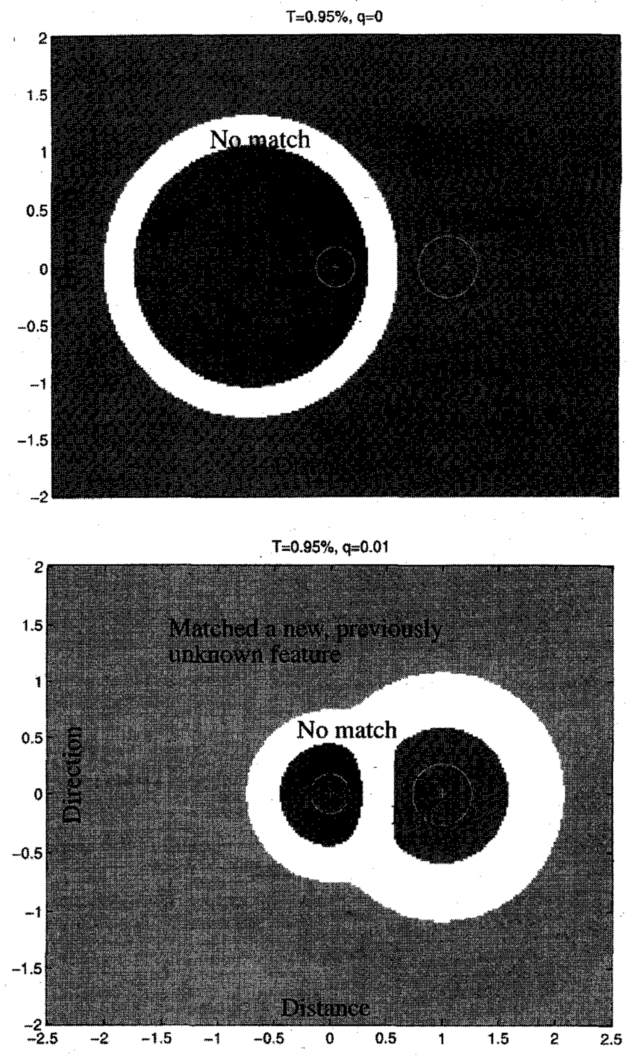

Fig. 3. Illustration of the association function and the effect of the parameter q. The plots shows the decision regions when matching one observation to one of two features. Note that this example uses unrealistic parameter values to better illustrate the principle. The variance of the estimate of feature two is twice that of feature one. In both plots a threshold T of 95% is used. The white regions indicate where no match is possible with this criterion. In the upper plot q is zero, indicating that the two features are the only ones possible. In the lower plot q is 0.01 indicating a small likelihood of detecting a previously unknown feature. Note the fundamental difference between the decision regions for these two cases! The features (at $(0,0)$ and $(1,0)$) are indicated by white plus signs and white circles with one standard deviation radius.

C. Simultaneous Association of Features

Thus far, the relations between the features coordinates have not been used. If a good map is available, either a priori or from earlier observations, but the robot position uncertainty is large, then the variance for each estimated feature will be large but some will be strongly correlated. In this case it is possible to get a higher probability for the hypothesis if more than one measurement is matched simultaneously. Thus we might recognize a corridor by its width or a room by its size rather than the by distance to the individual walls.

As an example, let us look at the case of matching a pair of two observations to a pair of two features in the state vector. The approach is the same as given earlier. The likelihood for Z_1, Z_2 being observations of features i and j is given by (see equation at the bottom of this page) and the association function for simultaneously associating Z_1 and Z_2 with feature i and j, respectively, is thus (see (12) at the bottom of this page).

In the derivation of (12) we have assumed that there is no correlation between the unobserved features. The normalizing denominator is partioned into three terms. The first term is the case where the measurements are observations of features estimated in the state vector. The second term consist of the sum of the likelihood functions for the case where one of the measurements is generated by a feature estimated in the state vector and the other measurements is an observation of a feature not yet estimated. The last term q^2 corresponds to the case where both measurements are new observations of previously unknown features.

D. Adding New Features to the State Vector

Those observation that could not be associated to a previously detected feature in the state vector might be observations of "new" features not yet included in $\hat{X}$. To avoid that Z_l introduces a duplicate we require that

$$1 - \sum_{i=1}^{M} A_i(Z_l, S_l) > T \tag{13}$$

where M is the number of features already included in $\hat{X}$. Thus, the approximate probability that the observation is not from one of the estimated features has to be greater than T for a new feature to be added.

When a new feature is added, the state vector is augmented with the two new variables and the covariance matrix with the 2×2 covariance matrix. The covariance between the newly added states and the other states is zero. The new state vector and covariance matrix in the filter is then given by

$$\begin{bmatrix} \hat{X} \\ Z_l \end{bmatrix} \quad \begin{bmatrix} \Sigma & 0 \\ 0 & S_l \end{bmatrix}.$$

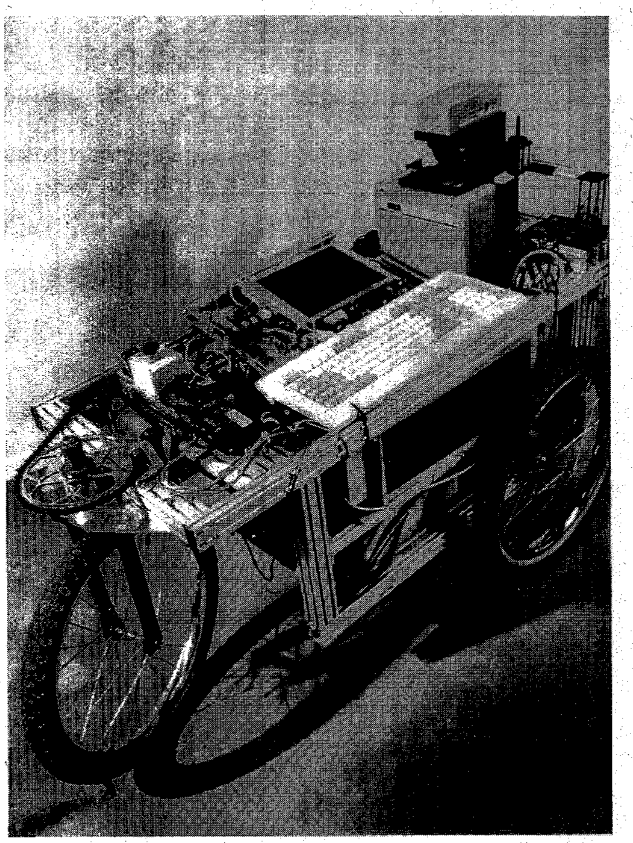

Fig. 4. LuSAR—*Lu*leå *S*emi *A*utonomous *R*obot. The two rear wheels have one motor each, with a third motor used for steering the single front wheel. Computation and control are performed by a standard industrial 486 PC running a multitasking operating system (QNX). Incremental encoders measure the rotation of the rear wheels and the steering angle. On the top of the robot is the scanning range measuring laser. The robot is capable of speed up to 2 m/s, but most tests were performed at speed below 1 m/s as higher speeds are impractical in the tight indoor environments.

E. Deleting Duplicate Estimates of a Feature

Even if the condition in (13) is fulfilled, there is a small probability that a new feature might be added by mistake, resulting in two separate estimates of the same feature. The

$$P_{ij}(Z_1, Z_2) = \frac{e^{-\frac{1}{2}\left(\left[\begin{smallmatrix} Z_1 \\ Z_2 \end{smallmatrix}\right] - \left[\begin{smallmatrix} H_i \\ H_j \end{smallmatrix}\right]\hat{X}\right)^T \left(\left[\begin{smallmatrix} H_i \\ H_j \end{smallmatrix}\right]\Sigma\left[\begin{smallmatrix} H_i \\ H_j \end{smallmatrix}\right]^T + \left[\begin{smallmatrix} S_1 & 0 \\ 0 & S_2 \end{smallmatrix}\right]\right)^{-1}\left(\left[\begin{smallmatrix} Z_1 \\ Z_2 \end{smallmatrix}\right] - \left[\begin{smallmatrix} H_i \\ H_j \end{smallmatrix}\right]\hat{X}\right)}}{(2\pi)^2\sqrt{\left|\begin{bmatrix} H_i \\ H_j \end{bmatrix}\Sigma\begin{bmatrix} H_i \\ H_j \end{bmatrix}^T + \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix}\right|}}$$

$$A_{ij}(Z_1, Z_2) = \frac{P_{ij}(Z_1, Z_2)}{\sum_{m_1=1}^{M} \sum_{\substack{m_2=1 \\ m_2 \neq m_1}}^{M} P_{m_1 m_2}(Z_1, Z_2) + \sum_{m=1}^{M}(P_m(Z_1) + P_m(Z_2))q + q^2} \tag{12}$$

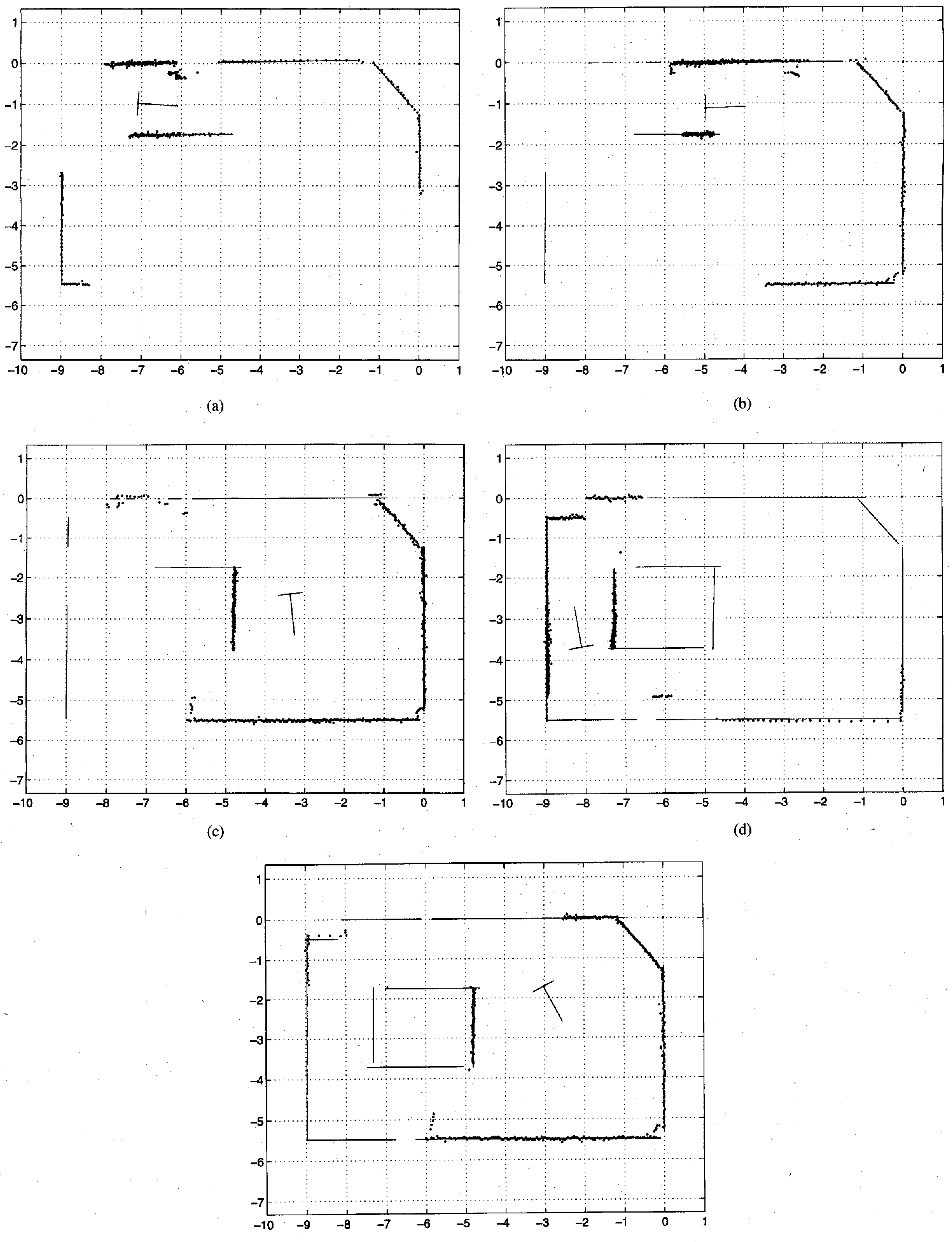

Fig. 5. Map generation test where the robot navigates around a rectangular object in the laboratory. The sequence of plots illustrate how the robot builds its map incrementally. The robot measures continuously while it moves, and made several measurements between each of the above plots. Each of the five plots show the robot position, the current state of the map and the latest range scan. (a) The first plot is taken before the robot started to move. Thus, all the walls in the map was generated from Hough peaks in the same range scan. (b) –(e). The subsequent plots show how the map is built as the robot moves around the laboratory. Note how the upper wall is rediscovered in plot (d). The final plot (e) shows the robot on its second loop around the room, successfully using the map it created on the first lap. Typical standard deviations are 2 cm. The endpoints of the lines are currently only recorded for plotting, and not filtered. The scale is in meters.

TABLE I
STATISTICS FROM TEN EXPERIMENTAL ROBOT RUNS AROUND THE LABORATORY IN FIG. 5. THE TABLE SHOWS THE SAMPLE STANDARD DEVIATION FOR THE SIZE OF THE ROOM, AND THE ANGLE BETWEEN THE WALLS. THE SECOND PART OF THE TABLE SHOWS THE MAXIMUM DEVIATIONS FROM THE MEAN RATHER THAN THE STANDARD DEVIATIONS

Standard deviations	After one lap		After two laps	
	Size	Angle	Size	Angle
Long walls	22 mm	5 mrad	15 mm	3 mrad
Short walls	17 mm	9 mrad	16 mm	7 mrad
Diagonal wall	—	14 mrad	—	9 mrad

Maximum deviations	After one lap		After two laps	
	Size	Angle	Size	Angle
Long walls	37 mm	7 mrad	24 mm	6 mrad
Short walls	28 mm	17 mrad	31 mm	15 mrad
Diagonal wall	—	32 mrad	—	15 mrad

highest risk for such duplicates is when the robot returns to the initial position and rediscovers features seen before—compare with Figs. 5 and 10. To detect such duplicates it is necessary to check that the estimates are well separated. This is done using the same principles as above.

Assume that $\hat{X}_M$ and $\hat{X}_i$ are both estimates of the same feature X_i, $i < M$. The estimated state vector is then

$$\hat{X} = \begin{bmatrix} I \\ H_i \end{bmatrix} \begin{bmatrix} X_1 \\ \cdots \\ X_{M-1} \end{bmatrix} + W$$

with $\mathrm{Cov}(W) = \Sigma$. The distance $\hat{X}_M - \hat{X}_i$ is used as a test variable in the modified association function

$$A_i^*(\hat{X}_M) = \frac{P_i^*(\hat{X}_M)}{\sum_{j=1}^{M-1} P_j^*(\hat{X}_M) + q} \tag{14}$$

where

$$P_i^*(\hat{X}_M) = \frac{e^{-\frac{1}{2}(\hat{X}_M - H_i\hat{X})^T([-H_i I]\Sigma[-H_i I]^T)^{-1}(\hat{X}_M - H_i\hat{X})}}{2\pi\sqrt{|[-H_i I]\Sigma[-H_i I]^T|}}.$$

If $A_i^*(\hat{X}_M) > T$ then the estimated state vector is merged to $2(M-1)$ elements using

$$\hat{X}^* = K^*\hat{X}$$

where

$$K^* = \left([I \ \ H_i^T]\Sigma^{-1}\begin{bmatrix} I \\ H_i \end{bmatrix}\right)^{-1} [I \ \ H_i^T]\Sigma^{-1} \tag{15}$$

with the covariance matrix

$$\Sigma^* = K^*\Sigma K^{*T}. \tag{16}$$

The merged state vector $\hat{X}^*$ and the covariance matrix Σ^* is then used as the new estimate of the feature parameters. Note that if there is no correlation between $\hat{X}_M$ and the other elements in $\hat{X}$, then (14) is identical to (11) and (15)–(16) can be rewritten in the form of a Kalman Filter update [15].

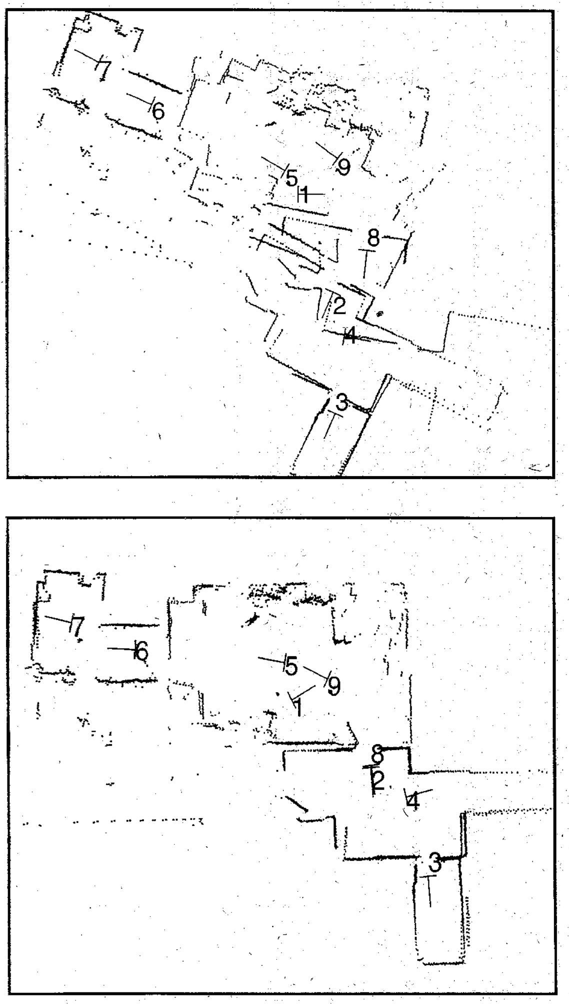

Fig. 6. The upper plot shows nine overlaid ranges scan aligned using dead reckoning. The lower plot shows the same measurements but aligned using position estimates from the navigation algorithm. During the test the robot passed between the rooms as indicated by the numbered "T" symbols. The final scan taken in position 9 during this test is also shown in Fig. 1. The figure size is about 15 by 10 meters.

V. EXPERIMENTAL RESULTS USING A RANGE MEASURING LASER ON A MOBILE ROBOT

The experiments were carried out using the mobile robot LuSAR (Luleå Semi Autonomous Robot), see Fig. 4. The tests described were for navigation around a large rectangular obstacle in a room and when re-entering a room three times with a disturbing slip over a tube in between. In both cases the robot returns to the initial position and rediscovers old features.

A. The Mobile Robot LuSAR

The algorithms were implemented on the mobile robot LuSAR (Fig. 4) equipped with a scanning range measuring laser. The laser, an IBEO Ladar, can measure distances up to 30 meters in a 270° sector with a resolution of 2 cm. Using the multitasking operating system QNX control tasks and odometric dead reckoning can run in parallel with the time-consuming measurement processing and classification.

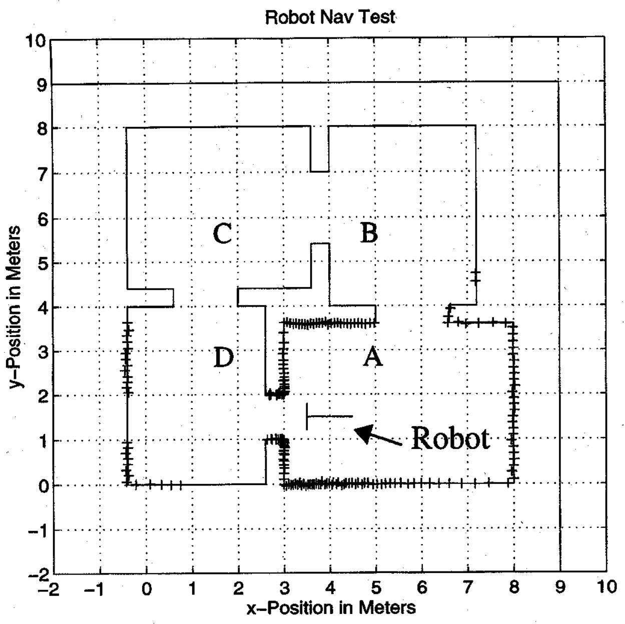

Fig. 7. In the simulations the robot moved between the four rooms,with noise added to both motion and measurements. The "+" signs are the random range measurements.

The robot is intended to be used as a semiautonomous, teleoperated robot controlled by fairly high-level telecommands. A human operator selects the sequence of commands while the robot performs low-level control and planning. With increasing autonomy, the communication bandwidth needed and the human workload will decrease. During the current mapping tests, the steering of the robot was by the operator. Examples of previously tested autonomous operations are passing through doors [7] and navigating along corridors [9]. The use of rate gyro was demonstrated in [10].

B. Building a Map of a Laboratory with an Obstacle

In this test a map is generated for a room with a rectangular object in the middle. The sequence of plots in Fig. 5 shows how the map was generated recursively as the robot was driven around the object.

To evaluate the accuracy of the map the robot was driven around the lab twice, saving the state (or map) after each lap. The speed varied between 0.5 and 1 m/s, generally higher during the second lap. This process was repeated twelve times. In two of these tests the robot made some small mistakes when generating the map; failing to include the short wall in the upper right corner of the map, and duplicating wall. Neither of these faults caused any serious disturbance to the system and are in no way fatal. The ten entirely successful runs were used to compute the statistics in Table I.

The results conform nicely with what could be predicted. The longer walls are estimated more precisely than the shorter ones, especially the angle, and the estimates improve after the second lap by about as much as can be predicted from theory. The standard deviations from the experiments agrees fairly well with those predicted by the Kalman filter.

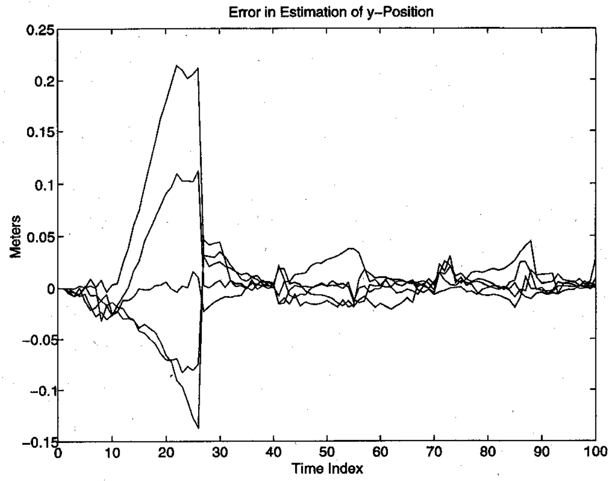

Fig. 8. The error in the distance to the lower wall as the robot moves around the four rooms in Fig. 7. The plot shows the error in the robot's y-position estimate during 5 different test runs.The robot builds the map during the first lap. It relocates the lower wall around time 26. During the following laps, the newly created map is used, giving much smaller errors.

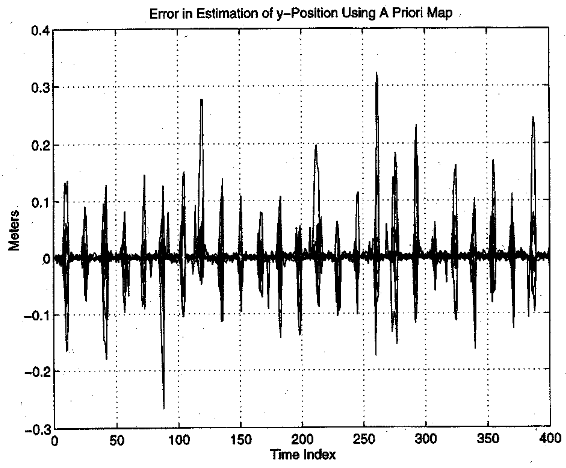

Fig. 9. The plot shows the error in the y-position of the robot for 10 simulations. In these simulations an *a priori* map was given. Each simulation corresponded to 12-and-a-half laps around the building modeled in Fig. 7.

C. Passing Through Several Cluttered Rooms

The next test was made with the robot passing through two laboratories, a hallway and a small room, Fig. 6. The robot started in one laboratory (pos 1), moved out into a hallway (pos 2 to 4), turned back into the laboratory (pos 5) and moved further into the next laboratory (pos 6, 7). From there it turned back through the first laboratory and into the hallway (pos 8) and back to the laboratory again (pos 9). (Positions refer to Fig. 6.) The plots show range measurements taken at the nine positions overlaid onto each other using the navigation algorithm, and for comparison using only dead reckoning from the odometers.

The rooms are rather cluttered and it can be quite difficult to detect the walls in the range scan. Still the robot was capable of navigating. The robot of course used many more scans to navigate than is shown in the plots, as it continuously scans

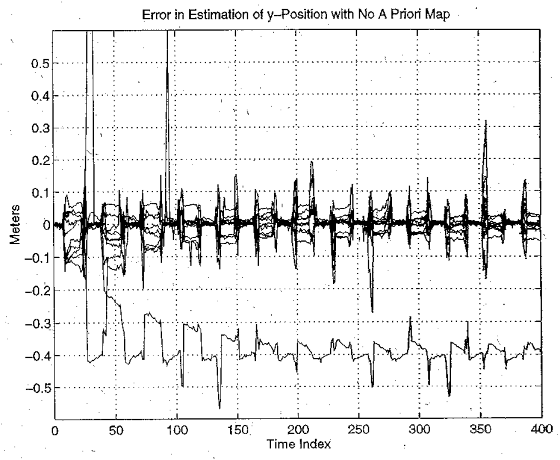

Fig. 10. Error in the y-position for 10 simulations identical to those in Fig. 9, except that no map was given in advance. Thus the errors are larger during the first loop and then decrease. A critical point is when the robot observes features that have been occluded for a long time. This first happens at time index 20. In one of the test runs the algorithm failed to associate the measurements of the wall used to calculate the y-position leading to a bias in the estimate.

the environment while it moves. The largest disturbance was caused when the robot drove over a tube (a rotation around $x = -4$ and $y = -3$).

VI. Simulations

In the simulations described here the robot moves between four rooms of varying sizes, Fig. 7. The robot moved in a circle between the rooms in the direction A B C D. The plots illustrates typical behavior of the algorithm under different circumstances, both with and without an *a priori* map. The association algorithm seems fairly reliable when the motion uncertainty is $<10\%$ of the travelled distance. It is illustrative to see how the covariance is reduced when the robot returns to its initial position Fig. 8.

A. Map Building and Association

In the first simulation the robot built its map while travelling between the four rooms. The state vector is initialized with the feature coordinates of two walls which are then used as the position reference to evaluate the performance of the navigation algorithm. Fig. 8 show the error in the y-position.

The map is created during the first lap. The following laps have about 5 times smaller errors as the map created during the first lap can now be used. The plot shows how the error increases during the first lap around the four rooms. When (around $t = 26$) the robot re-locates the lower wall, the error is immediately decreased as the algorithm successfully associates the new measurement with the correct feature in the map. During the following loops the error is much smaller as the robot can use the map it created during the first loop.

B. Navigating with and Without an A Priori Map

An *a priori* map can be provided to the robot by initializing the state vector with the feature coordinates for all the walls. This section compares the behavior of the algorithm with and without such an a priori map.

Noisy odometric readings were simulated having the following statistics for one motion step.

$$\text{Mean}(\hat{U}) = \begin{bmatrix} 0.46 \\ 0.047 \\ 0.20 \end{bmatrix}$$

and

$$\text{Cov}(\hat{U}) = \begin{bmatrix} 0.0025 & 0.0002 & -0.0003 \\ 0.0002 & 0.0023 & 0.0097 \\ -0.0003 & 0.0097 & 0.042 \end{bmatrix}$$

with translations in meters and rotations in radians.

In Fig. 9, the error in the estimated y-position is plotted when the robot navigated using an *a priori* map. The y-position is defined here as the distance to the lower wall. The error, as seen in the plot, increases when robot is between rooms A and B and between rooms C and D. In these locations, the laser receives few measurements from the walls parallel to the x-axis. Therefore the RWHT only extracts feature coordinates corresponding to walls parallel to the y-axis. Hence there is no reduction of the "temporary dead reckoning" error in the y position. The orientation (not plotted) is, however, still being updated.

To test the map building algorithm, the simulations were repeated without an a priori map. Fig. 10 shows the errors in the estimate of the y-position. Comparing this with the case of an a priori map (Fig. 9) we see that the peaks in the estimation errors appear at the same positions. The difference is that the error is significantly larger when the robot is in rooms B or C where the wall defining the y-axis is occluded. As the robot moves the map is improved. Therefore the error in the estimated y-position at time index 400 is smaller than at time index 15 (the same spatial position of the robot).

The first loop when the map is generated is the most critical period. In one of the test runs, the algorithm failed to associate the measurements from the wall which was used to calculate the y-position. This resulted in the lower of the curves in Fig. 10.

VII. Conclusion and Future Work

The algorithm was tested in several other experiments not reported here. Reliable results were obtained even in *cluttered* environments. In a well structured environment, the standard deviation of the error was about 2 cm for range and 0.3° for orientation (Table I). The reliability of the system can be exemplified by the fact that it continued to function even when, due to a mechanical fault, one rear wheel was spinning freely on its axis. Simulations were made to study failures for large motion uncertainty. The algorithm seems reliable if the motion uncertainty is less than 10% of the travelled distance.

The principle drawback at present is the *computation time.* For small maps, with up to 25 walls, real-time operation is feasible with updating times of 1–2 s on a 33-MHz 486-PC. As the size of the map increases the time increases quickly to unacceptable levels. A solution is to use approximations of the distributions and explore only a subset of the combinations.

This will also allow simultaneous matching of more than two features, greatly increasing the robustness of the algorithm. Another method to improve speed is to represent states that are far away from the robot relative to other states, rather than directly relative to the robot.

The reader should observe that only distances and angles to walls are used. This keeps the results unified. It is obvious that further improvements such as including additional features like "jump edges" at range discontinuities, isolated objects and corners would be straight forward to implement. Improved estimation of the endpoints of lines will make it possible to use the map for path planning. Matching of lines might also use the endpoints, either as a consistency test, or more closely incorporated in the probability estimation. This will be the subject of future developments

The navigation system has been used both in the semi autonomous robot LuSAR and in an autonomous plastering robot tested on a construction site.

REFERENCES

[1] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall, 1979.
[2] N. Ayache and O. D. Faugeras, "Maintaining representations of the environment of a mobile robot," *Autonomous Robot Vehicles*. Berlin: Springer-Verlag, 1990.
[3] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. New York: Academic Press, 1988.
[4] T. J. Broida, "Kinematic and statistical models for data fusion using Kalman filter," *Data Fusion in Robotics and Machine Intelligence*. New York: Academic Press, 1992.
[5] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[6] A. Elfes, "Sonar-based real-world mapping and navigation," *Autonomous Robot Vehicles*. Berlin: Springer-Verlag, 1990.
[7] J. Forsberg, U. Larsson, and Å. Wernersson, "Mobile robot navigation using the range-weighted Hough transform," *IEEE Robot. & Automation Mag., Special Issue on Mobile Robots*, vol. 2, no. 1, pp. 18–26, Mar. 1995.
[8] J. Forsberg, U. Larsson, P. Åhman, and Å. Wernersson, "The Hough transform inside the feedback loop of a mobile robot," in *IEEE Conf. Robot., Automat.*, 1993, pp. 791–798.
[9] ——, "The Hough transform inside the feedback loop of a mobile robot: Navigating in corridors," in *SSAB Symp. Image Processing*, Uppsala, Feb. 1992.
[10] T. Högström, T. Hultberg, and Å. Wernersson, "A semi autonomous robot with rate gyro supported control and a video camera," in *IFAC Int. Workshop Intell. Autonomous Vehicle*, Southampton, 1993, pp. 25–30.
[11] U. Larsson, C. Zell, K. Hyyppä, and Å. Wernersson, "Navigating an articulated vehicle and reversing with a trailer," in *IEEE Conf. Robot., Automation*, 1994, pp. 2398–2404.
[12] J. Leonard, H. Durrant-Whyte, and and I. J. Cox, "Dynamic map building for an autonomous mobile robot," in *IROS-90*, pp. 89–95.
[13] P. Moutarlier and R. Chatila, "Stochastic multisensory data fusion for mobile robot location and environment modeling," in *Robot. Res. 5th Int. Symp.*, pp. 85–94.
[14] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous Robot Vehicles*. Berlin: Springer-Verlag, 1990.
[15] L. Söderström and P. Stoica, *System Identification*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

Ulf Larsson received the M.S. degree in electrical engineering from Luleå University of Technology in 1992. He is currently pursuing the Ph.D. degree in robotics at Luleå University of Technology. His research interests are in the area of sensing and navigation for mobile robots.

Johan Forsberg received the M.S. degree in electrical engineering in 1992 and Licentiate of technology in 1995, both at Luleå University of Technology. He is currently pursuing the Ph.D. degree in robotics at Luleå University of Technology. His research interests are in the area of sensing and navigation for mobile robots.

Åke Wernersson (S'69–M'75) receievd the M.S. degree in engineering physics from Chalmers and the Ph.D. degree in optimization and system theory from the Royal Institute of Technology, Stockholm, Sweden. He is a Docent in Automatic Control, and now professor in robotics and automation at Luleå University of Technology. he spent three years with SAAB Avionics and eight years with FOA (Radar Systems and Information Technology). His current research is on models and algorithms for extracting geometric shapes, and features and motion using noncontact sensors (laser, range camera, coherent ultrasonic). Typical applications are sensor feedback for robot assembly, curve and surface following, vehicle navigation, inspection and intervention robots, and the new exiting area of telerobotics.