

Pose Tracking Using Laser Scanning and Minimalistic Environmental Models

Patric Jensfelt, *Member, IEEE*, and Henrik I. Christensen

Abstract—Keeping track of the position and orientation over time using sensor data, i.e., *pose tracking*, is a central component in many mobile robot systems. In this paper, we present a Kalman filter-based approach utilizing a minimalistic environmental model. By continuously updating the pose, matching the sensor data to the model is straightforward and outliers can be filtered out effectively by validation gates. The minimalistic model paves the way for a low-complexity algorithm with a high degree of robustness and accuracy. Robustness here refers both to being able to track the pose for a long time, but also handling changes and clutter in the environment. This robustness is gained by the minimalistic model only capturing the stable and large scale features of the environment. The effectiveness of the pose tracker will be demonstrated through a number of experiments, including a run of 90 min, which clearly establishes the robustness of the method.

Index Terms—Laser scanner, localization, minimalistic models, pose tracking, sensor modeling.

I. INTRODUCTION

THERE are many challenging problems which have to be solved before we have all the methods needed for deployment of robots in ordinary domestic environments. One of them is localization. Localization is the process of finding the position and orientation of the robot, i.e., the pose. Knowing the correct pose of the robot is a condition for the successful completion of many of the tasks which a robot might be required to perform.

Localization can be thought of as consisting of two parts, initialization and maintenance. Initialization is often referred to as global localization and consists of finding the pose of the robot without any prior knowledge but a map. Building the map can also be viewed as part of the initialization. Once the pose is found, localization becomes the task of maintaining the estimate of the pose, i.e., tracking the pose. In this paper, we will focus on the latter, that is, *pose tracking*.

Much effort has been spent on modeling the world very accurately, to achieve better performance. We chose to turn this upside down and instead ask ourselves how simple can the map be and still be useful for solving the localization problem. Therefore, one design criteria is to use as simple a model as possible. A simple model has the potential of being very robust over time

and provide means for having an efficient tracking system that fits well in an autonomous system with limited computational resources. The only requirement we pose on the localization system is that it must not rely on altering the environment in any way. This requirement rules out methods which rely on, e.g., bar codes or active beacons.

A. Paper Outline

In Section II, related work on pose-tracking and localization is discussed. The problem is defined and the assumptions made are discussed in Section III. Section IV describes the overall structure of the algorithm. A model for the odometry is developed in Section V, and Section VI is devoted to characterization of the laser scanner used and the method for extraction of environmental features. Section VII contains experimental results and conclusions, and a discussion can be found in Section VIII.

II. RELATED WORK

Given that localization is such an important component in any mobile robot system, it is not surprising that many different approaches have been suggested in the literature. In this paper, we will only briefly mention some of them, and we instead refer to, for example, [1] for a more thorough survey. We conclude that there are two prevailing categories of approaches, depending on the modeling technique being used. These are *parametric methods* and *grid-based methods*.

A. Parametric Methods

The idea of trying to extract features from the environment is quite natural. Examples of highly specific features would be labels on doors which specify the room it is leading to. In structured environments, such as most office areas, lines, corners, and edges are common features. The features can be parameterized by, e.g., their color, length, width, position, etc. Leonard *et al.* [2] are quite firm in their conviction that feature-based methods are superior, when they say “*we believe that navigation requires a feature-based approach in which a precise, concise map is used to efficiently generate predictions of what the robot should see from a given location.*” The Kalman filter is a key component in most implementations of parametric methods, providing a good setting for pose prediction and sensor fusion.

An extreme example of feature-based localization is the work by Christensen *et al.* [3] where a complete CAD model of the environment is used for localization and pose tracking through feature matching and three-dimensional recovery using stereo vision.

Manuscript received June 1, 2000; revised January 20, 2001. This paper was recommended for publication by Associate Editor G. Hager and Editor S. Hutchinson upon evaluation of the reviewers' comments. This work was supported by the Swedish Foundation for Strategic Research through the Centre for Autonomous Systems. This paper was presented in part at the International Conference on Robotics and Automation (ICRA), Detroit, MI, 1999.

The authors are with the Centre for Autonomous Systems, Royal Institute of Technology, SE-100 44 Stockholm, Sweden (e-mail: patric@s3.kth.se).

Publisher Item Identifier S 1042-296X(01)04805-4.

B. Grid-Based Methods

The parametric methods have the disadvantage that an explicit model is needed for all the information that is used. Another thought is to divide the work space into a grid where each cell in the grid represents a part of the world. One advantage of this approach compared to the parametric is the following as Hager and Mintz [4] points out: “*The grid-based method is only an approximative solution, but it is much less sensitive to assumptions about the particular form of the sensing system.*” Schiele and Crowley [5] conclude that the results using grids “*are comparable or more accurate than the ones we obtain with previous work using a parametric model of segments extracted directly from sensor data.*” Moravec and Elfes made the grid-based techniques popular with their paper [6] in 1985 where the occupancy grid is presented.

Burgard *et al.* also use an occupancy grid representation for the map. In [7], the knowledge of the robot’s position is stored in another grid, a position probability grid, where the cells contain the probability of the robot being at that pose. The method has proven successful, but requires considerable amount of computational power. In [8], inspired by work in other research areas, a sample set is used to represent the pose knowledge. Both approaches provide the means to do both pose tracking and global localization.

III. PROBLEM DEFINITION AND ASSUMPTIONS

As was seen in Section II, the problem of pose tracking is well studied in the literature and further efforts must be weighed against the potential gain. In a system operating in a real world setting, where localization is only a small, but yet important, part of the whole integrated system, the computational resources are limited. This, together with the fact that several tasks must be performed simultaneously, e.g., localization, planning, and obstacle avoidance, means that we should aim for low computational complexity. Artificial landmarks are often used to reduce the complexity of the localization problem. Here we do not allow any artificial landmarks or any other form of engineering of the environment. The reason for this is that the system should be able to operate in a typical domestic or office environment.

Most robot platforms are equipped with wheel encoders, *odometry*, that can be used to measure relative motion. This means that if the initial pose is given, odometry can be used to keep track of the pose, under ideal conditions. Such is not the case in reality, where, for example imperfections in the kinematic model of the robot and wheel slippage will cause the pose estimation error to grow without bounds. Still, odometry is a resource that should be utilized to make the problem of localization simpler, as it is known to be very reliable over short distances under normal circumstances.

To bound the error in the pose estimation, external sensors must be used. These sensors can provide information about the absolute position of the robot by associating measurements with parts of the map. Sonar sensors have been used extensively in localization research. The sonar sensor is reliable and cheap, but it has clear limitations in its use. Due to the wide beam width, it requires heavy post-processing to extract the important information. The experimental platform under consideration is equipped

with a scanning laser sensor, also providing range data, but with much higher angular resolution. Computation power can thus be saved at the cost of a more expensive sensor. To make this decision, it is important that the increase in cost is justified. Even if the SICK laser scanner is more than an order of magnitude more expensive than the sonars today, the price will drop when the market for it increases. This will make the arguments for using sonar sensors even weaker. The reason why we do not even consider vision as a sensor for solving the localization problem is the lack of robustness and the computational cost of present vision algorithms. We do however acknowledge the large potential of vision.

A. Environmental Model

To realize a low complexity pose tracking system, the choice of environmental model is of course of great importance. We believe that a parametric method is the best way to fully utilize the simplicity of the model and propose to use a rectangular model for each room. The choice of the rectangular model is a design decision, aiming to capture only the most dominant and stable features in the environment. This choice would of course not be motivated in an outdoor setting, but as we here concentrate on an indoor environment we find that it is justified. The main benefit of the rectangular model is that it is very likely to be robust over time. The more details that are used in the environmental model, the less time it is typically valid. As the sides of the rectangle correspond to walls, they are highly robust over time. Lines were chosen over, e.g., points since points are more likely to move than large scale lines. Points associated with wall corners meet our requirements, but most points in the environment stem from movable objects. Note that we do not assume that the lines are parallel to x or y axes and also allow for the case that nonrectangular models may be required for some rooms. One more very clear advantage with using large scale structures, such as lines, is there are not that many of them, which makes the data association easier.

Let \mathcal{M} denote the environmental model which is a set of lines m_i , i.e.,

$$\mathcal{M} = \{m_i, i = 1, \dots, |\mathcal{M}|\}.$$

m_i is described by its start point (x_s, y_s) and end point (x_e, y_e) , which gives it a direction. The direction is used to define from which side the line can be seen. By our definition a line can be seen if the direction of the line is from right to left when we look at it.

B. Estimation

Given that we know the initial pose of the robot, a good approximation of the robot at every time instant can be calculated using odometric information. Knowing the approximate pose of the robot also makes the data association problem considerably easier. Sensor data that is likely to belong to the walls (the sides of the rectangle model) can be extracted effectively, and be used for updating the pose estimate and thus bound the estimation error.

There are many ways to mathematically handle a problem of this nature. It is clear that it is an estimation problem and in

some sense an optimization problem. How do we incorporate the information that we get from the sensors, to calculate the best possible estimate of the pose? In order to answer this, we need to specify, best in what sense? As is often the case, we chose to define best in the least square sense. The use of least square estimation is justified by the fact that outliers are efficiently rejected through use of narrow validation gates that reject noisy data prior to the model matching. We will furthermore assume that the probability density function for the robot's pose can be represented by a unimodal function, that is, one having a single maximum. This puts high demands on the data association. In certain situations, a large uncertainty in the pose of the robot might introduce ambiguities when associating data, leading to a need for a multimodal description of the probability distribution. Given that the estimate is updated several times per second, it is however reasonable to assume unimodality due to the use of a sparse model of the environment.

C. State of the System

We assume the state of the system to be the pose of the robot, $\mathbf{x} = (x, y, \theta)^T$. This means that we assume the world to be stationary, or at least everything that is not stationary is to be considered as noise, and as such cannot be used for localization.

IV. ALGORITHM

To perform the least square estimate of the robot pose in real time, we need a recursive algorithm. The alternative would be to keep all the information in a batch and do a complete calculation at every time instant. The price we pay for real-time performance is that once a piece of information has been used and is incorporated into the estimate of the pose, it is lost forever. That is, we cannot go back and reconsider a decision concerning, for example, data association.

We will use the Kalman filter framework that has been used by numerous researchers (see, e.g., [9], [2], and [10]–[14]) and proven to provide a good solution for sensor fusion. The Kalman filter will give an optimal estimate of the pose given the information at hand, assuming that the model of the system is correct and that all sources of noise are Gaussian. The Gaussian assumption is normally not fulfilled, which yields a suboptimal estimate. Despite the non-Gaussian nature of most real world noises, the Kalman filter is still reported to perform very well [2], [11], [12].

At our disposal for tracking the pose of the robot, we have information from odometry and the laser sensor. The odometry can be used for short-time predictions of relative motion and the 180° scan from the laser sensor in combination with our environmental model give evidence about the absolute pose of the robot. The overall algorithm is shown in Fig. 1.

A. State Prediction Using Odometry

Let $\hat{\mathbf{x}}_k$ denote the estimate of the state \mathbf{x} at time k . Let $\mathbf{f}(\mathbf{x}, \mathbf{u})$ describe how the robot moves given the input \mathbf{u} . This function is typically nonlinear and associated with some uncertainty. The

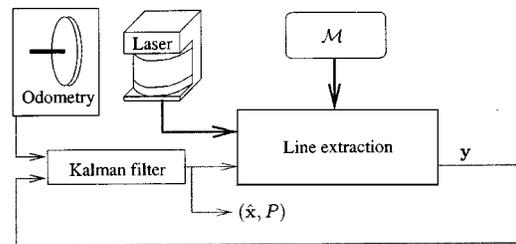


Fig. 1. The algorithm.

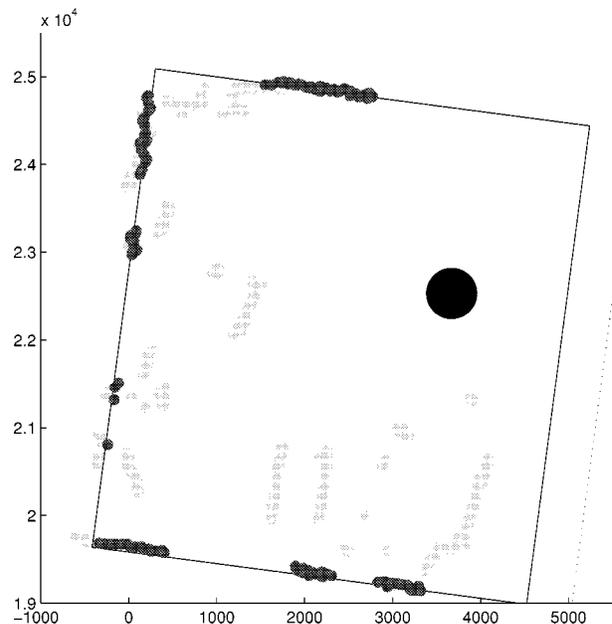


Fig. 2. Normally only two and in some cases three walls in a room is in the field of view of the laser scanning over 180°.

state of the system can thus be modeled as evolving according to

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k \quad (1)$$

where \mathbf{w}_k is noise, capturing the uncertainty of the odometric model. This noise is assumed white and Gaussian. See Section V for a more in-depth discussion about the function $\mathbf{f}(\mathbf{x}, \mathbf{u})$.

B. Feature Description and Selection

To describe the lines associated with the rectangular model extracted from laser data, we use (ρ, α, l) , where ρ is the perpendicular distance to the line, α is the orientation of the line, and l is the length of the line (see Fig. 3). The length of the line l is used to reduce the risk of making errors in data association. We only consider line measurements that fulfill $l \geq l_{\min}$ (in our case $l_{\min} = 1$ m).

Fig. 2 shows a scene from one of the rooms in the environment under consideration. As can be seen, the rectangular model only captures the very large scale structure of the room and in most cases offers only two features for tracking the pose. In Fig. 2, these two features are the lower and the left wall. The upper wall, which is also partly visible, results in a too short measurement ($l < l_{\min}$).

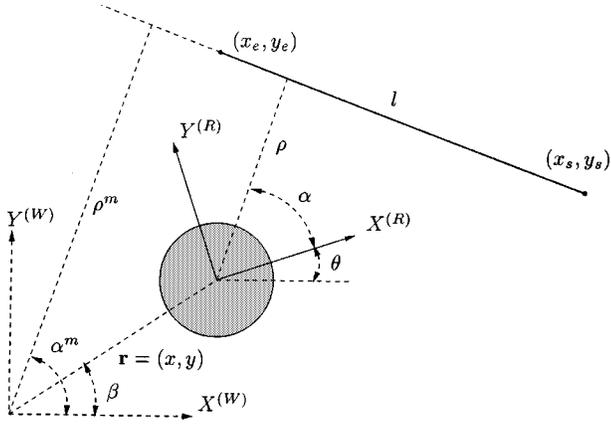


Fig. 3. The parameters defining a line.

The laser scanner is sensitive to occlusions as all laser beams stem from a point and thus an obstacle placed in front of the sensor will render it more or less blind. Therefore, visible features in neighboring rooms will also be considered for tracking purposes. We have limited us to lines from neighboring rooms as this has proven to be enough. The lines are stored in each room and thus we do not have to loop through all lines every time to determine what lines to consider. A line in the current room is visible if it fulfills the following criteria.

- $\text{AngleDiff}(\varphi_e^{(S)}, \varphi_s^{(S)}) > 0$ (direction condition)
- $(0 \leq \varphi_s^{(S)} \leq \pi) \vee (0 \leq \varphi_e^{(S)} \leq \pi)$ (in field of view), where φ_s and φ_e are the angles to the start and end point of the line, respectively, and $\text{AngleDiff}(\varphi_1, \varphi_2) \in (-\pi, \pi]$. The super script (S) refers to sensor coordinates. A line in a neighboring room is visible if it can be seen through the door leading to that room, i.e., as follows.
- $\text{AngleDiff}(\varphi_e^{(S)}, \varphi_{dr}^{(S)}) > 0 \wedge (\text{AngleDiff}(\varphi_{dl}^{(S)}, \varphi_s^{(S)}) > 0)$ where $\varphi_{dl}^{(S)}$ and $\varphi_{dr}^{(S)}$ are the angles to the left and right door post, respectively. Additional constraints will have to be added to handle nonconvex polygonal line models. For the pose update in a Kalman filter framework, we need to predict the parameters of the line to be extracted from the current state of the robot and the environmental model. Introduce $\mathbf{h}_i(\mathbf{x}, \mathcal{M})$ as the measurement function, i.e., $\mathbf{y}_i = (\rho_i, \alpha_i)^T = \mathbf{h}_i(\mathbf{x}, \mathcal{M}) + \mathbf{v}_i$, where \mathbf{v}_i is the measurement noise. We here let $i = [1, \dots, M]$, where M is the number of measurements. With the same notation as in Fig. 3 we get

$$\mathbf{h}_i(\mathbf{x}, \mathcal{M}) = \begin{pmatrix} \rho_i^m - \sqrt{x^2 + y^2} \cos(\beta \alpha_i^m) \\ \alpha_i^m - \theta \end{pmatrix} \quad (2)$$

where ρ_i^m is the distance to the line from the origin of the world coordinate system and α_i^m is the corresponding angle, i.e., $\mathbf{m}_i = (\rho_i^m, \alpha_i^m)^T$. This function can be expressed as the linear term $\mathbf{y}_i = H_i \mathbf{x} + \mathbf{m}_i$. H_i is given by

$$H_i = \begin{pmatrix} H_{11} & H_{12} & 0 \\ 0 & 0 & -1 \end{pmatrix} \quad (3)$$

where

$$H_{11} = -\frac{x}{r} \cos(\beta - \alpha_i^m) - \frac{y}{r} \sin(\beta - \alpha_i^m)$$

$$H_{12} = -\frac{y}{r} \cos(\beta - \alpha_i^m) + \frac{x}{r} \sin(\beta - \alpha_i^m)$$

and $r = \sqrt{x^2 + y^2}$.

C. Extended Kalman Filter

As the odometric model is nonlinear, the standard Kalman filter equation cannot be applied, instead the extended Kalman filter has to be used. The extended Kalman filter is typically divided into two parts, often referred to as *time-update* and *measurement-update*. If we let $\hat{\mathbf{x}}$ denote the estimate of the state \mathbf{x} , and P for the covariance matrix, the *time-update* can be written

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k) \quad (4)$$

$$P_{k+1|k} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} P_{k|k} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^T + Q_k \quad (5)$$

where $\hat{\mathbf{x}}_{k+1|k}$ should be interpreted as the pose estimate at time $(k+1)$ using sensor data up to and until time k . Q_k is the process noise matrix, capturing the uncertainty in the odometric model, $Q_k = E[\mathbf{w}_k \mathbf{w}_k^T]$.

As all lines are extracted from the same laser scan, the measurements are correlated and hence there is a nonzero covariance between the measurement noises \mathbf{v}_i . To handle the measurements correctly, this covariance must be accounted for. To reduce the computational burden though, we have chosen to neglect this covariance. By doing so, the *measurement-update* can be done sequentially. Let $\hat{\mathbf{x}}_{k+1|k,0} = \hat{\mathbf{x}}_{k+1|k}$ and $\hat{\mathbf{x}}_{k+1|k,\max(\mathcal{V})} = \hat{\mathbf{x}}_{k+1|k+1}$, with the corresponding notation for the state covariance matrix P . The measurement update can then be done by looping over all $i \in \mathcal{V}$ (all visible features) and performing

$$K_{k+1,i} = P_{k+1|k,i-1} H_i^T (H_i P_{k+1|k,i-1} H_i^T + \Lambda_i)^{-1} \quad (6)$$

$$\hat{\mathbf{x}}_{k+1|k,i} = \hat{\mathbf{x}}_{k+1|k,i-1} + K_{k+1,i} (\mathbf{y}_i - h_i(\hat{\mathbf{x}}_{k+1|k,i-1}, \mathcal{M})) \quad (7)$$

$$P_{k+1|k,i} = (I - K_{k+1,i} H_i) P_{k+1|k,i-1} \quad (8)$$

where Λ_i is the measurement noise (discussed in Section VI-D). The complexity of the Kalman filter update is linear in the number of measurements. The algorithm as a whole is also linear in the number of lines that are visible in each step as we, for each of these lines, try to extract a corresponding line from the laser scan.

V. ODOMETRIC MODEL

As with all sensors, a model of the odometry provides valuable information about performance and limitations. The aim is a model that can be used in an iterative update procedure, such as the Kalman filter. The model should provide an estimate of the robot motion as well as the uncertainty in this estimate. It is also desirable that the model be consistent in the sense that it should give the same result independent of how the path is

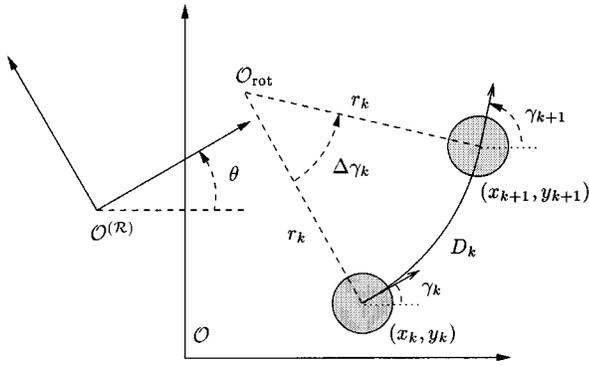


Fig. 4. The motion of the robot is approximated to be on arcs.

segmented. In [15], such a model is developed for a differential drive robot. Here, as well as in [15], we consider robot motion along arcs (see Fig. 4).

Let $\mathbf{u}_k = (D_k, \Delta\gamma_k)$ be the input to the odometric model, where D_k is the distance traveled along the arc and $\Delta\gamma_k$ is the change in motion direction. With this notion the radius of the motion is given by

$$r_k = \frac{D_k}{\Delta\gamma_k}. \quad (9)$$

Note that $\gamma_k = \theta_k + \phi_k$ where ϕ_k is the steer angle in robot coordinates, measured by the encoders and assumed to be known with negligible errors. A negative r_k corresponds to turning clockwise and a positive r_k corresponds to turning counter-clockwise. The odometric model can now be written

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) = \begin{cases} \begin{pmatrix} x_k + \frac{D_k}{\Delta\gamma_k} (\sin(\gamma_k + \Delta\gamma_k) - \sin \gamma_k) \\ y_k - \frac{D_k}{\Delta\gamma_k} (\cos(\gamma_k + \Delta\gamma_k) - \cos \gamma_k) \\ \gamma_k + \Delta\gamma_k \end{pmatrix}, & |\Delta\gamma_k| > 0 \\ \begin{pmatrix} x_k + D_k \cos \gamma_k \\ y_k + D_k \sin \gamma_k \\ \gamma_k \end{pmatrix}, & \delta\gamma_k = 0 \end{cases} \quad (10)$$

We assume that the errors of the two components of the odometric input \mathbf{u}_k are uncorrelated.

VI. LINE EXTRACTION

Given a set of range readings known to originate from a modeled line, it is easy to extract the line parameters. However, before this can be done the data association problem must be solved. Assuming we have an estimate of the robot pose, the pose of a modeled line can be predicted. The design goal is therefore twofold as follows: 1) classify each point as belonging to a particular model line or as being an outlier and 2) estimate the parameters of the lines. For this task we propose to use a two-step approach, centered around two different line extraction algorithms in combination with validation gates (see Fig. 5). The first line extraction algorithm is robust against outliers, but provides only limited accuracy. The second step provides accuracy

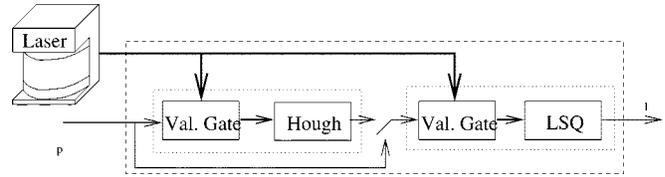


Fig. 5. Using the predicted pose of the robot in combination with map information, validation gates can be defined. The first step contains a local range weighted Hough transform fed with data via a validation gate. The second step is a least square method using data from a second validation gate.

assuming that the input data is “clean.” When the pose uncertainty of the robot is small, the first step can be bypassed. In the following sections, the different parts of the line extraction algorithm are described in some detail.

A. Laser Characteristics

Before designing a method for extracting lines from sensor data, it is important to know the underlying characteristics of this data. The sensor we use is a PLS laser scanner from SICK electro-optics. It gives 361 range readings, $\mathbf{z}_i = (r_i, \phi_i)$, covering a 180° field of view.

The range readings are quantized with step $\Delta r = 50$ mm. Fig. 6 shows a part of a scan taken against a wall approximately 5.5 m away from the sensor. The discretization is clearly visible in the right subfigure, where the same data is shown with different scaling and circular arcs with 50-mm spacing. Due to the gross discretization it is not easy to get an estimate of the underlying range distribution, i.e., given that the true range is R , what is the probability distribution function for measuring r . To get a better view of the underlying range distribution, we perform an experiment where 100 range readings, $q_{i,j}, i = 1, \dots, 100$, are collected at each of 50 ($j = 1, \dots, 50$), tightly spaced positions in front of a wall. Let Δd_j ($j = 1, \dots, 50$) be the distance traveled toward the wall according to odometry. Fig. 7 shows a histogram over $q_{i,j} + \Delta d_j$. Assuming the odometry to have negligible uncertainty over the short traveled distance, the spread in the figure is due to the error distribution inherent in the laser scanner. This distribution clearly resembles a Gaussian. Its standard deviation can be estimated to ≈ 24 mm. The resulting Gaussian is overlaid in Fig. 7. As the distance measurements are quantized, only multiples of the quantization step can be measured, $n\Delta r, n = 0, 1, 2, \dots$. Assuming R to be the true distance, the pdf for the measurements can be written

$$p(n\Delta r|R) = \frac{e^{-(1/2)(n\Delta r - R/\sigma_r)^2}}{\sum_m e^{-(1/2)(m\Delta r - R/\sigma_r)^2}}$$

that is a sampled Gaussian. The variance in a measurement error ($R - n\Delta r$) will, because of the quantization, depend on the true distance according to $V(R - n\Delta r|R) = \sum_n (R - n\Delta r)^2 p(n\Delta r|R)$. Fig. 8 shows the corresponding standard deviation as a function of true distance R . We will make the simplifying assumption that the standard deviation for the measurement error is independent of the true distance and equal to the worst case, i.e., approximately 26 mm.

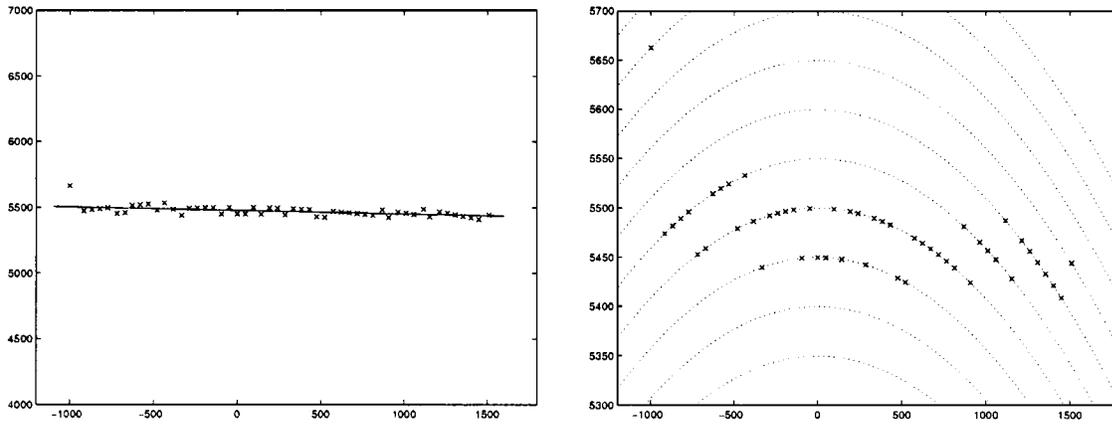


Fig. 6. Segment of a scan taken of a wall ≈ 5.5 m away from the sensor. A line marking the location of the wall has been added to the left subfigure. The right subfigure shows the range readings along with arcs with 50-mm spacing, clearly showing the discretization.

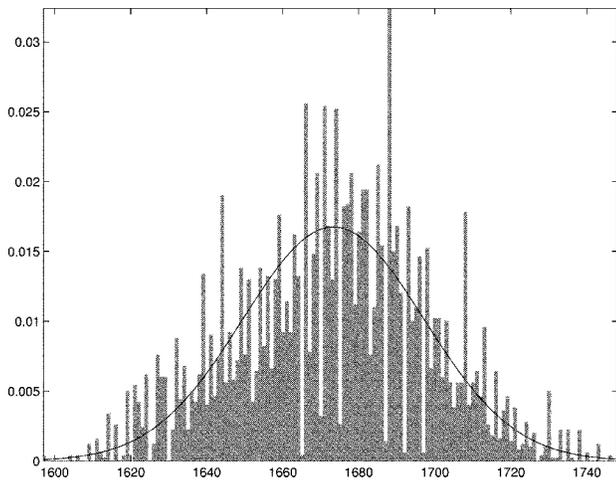


Fig. 7. The middle range reading from 100 scans taken at 50 closely spaced distances from a wall. The distance traveled toward the wall has been added to the range to get an estimate of the underlying distribution. The distribution is approximated as Gaussian with a standard deviation of ≈ 24 mm.

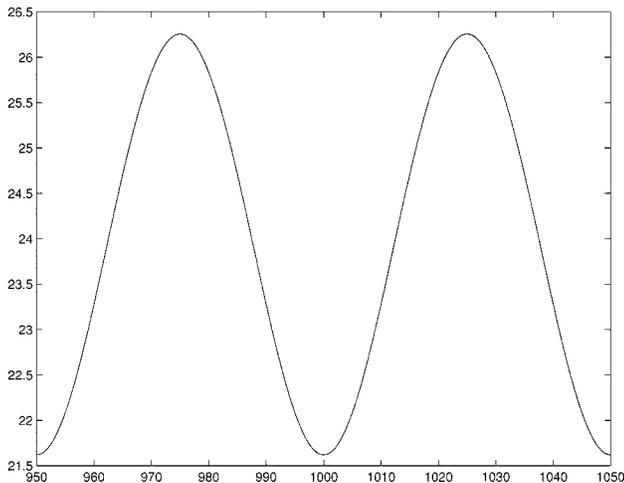


Fig. 8. Standard deviation for measurement error as a function of true distance.

The laser energy propagates in a cone, just like the ultrasonic energy of a sonar sensor. The difference is in the beam width. The beam width of the laser is less than a degree whereas the standard Polaroid sensor has a full beam width of approximately

25° . To be able to determine the beam width and hence the size of the footprint, we performed a series of experiments. The laser sensor is placed approximately 4.7 m away from two wooden boards. By sliding the boards perpendicular to one laser beam and monitoring the measurement, the size of the beam can be estimated. When the boards are not detected, the measured distance will be the distance to the wall behind.

The boards can be separated approximately 20 mm and still be detected. This separation at a distance of 4.7 m is equivalent to a beam width of approximately 0.25° . The size of the beam that is capable of detecting that there is something in front of the wall, but not necessarily correctly measure the distance, is approximately 0.43° . That is, if only a small fraction of the beam is reflected by an object, a phantom measurement might arise with a distance corresponding to a nonexistent object at a distance somewhere between the closest object and the one behind. As the laser beams are separated by 0.5° , there is a blind region between the beams.

Now that we know the characteristics of the measurements we formulate our model. We will disregard the phantom measurement effect and assume the error in angle to be independent of the error in range. Furthermore we assume the error in angle to be Gaussian. The i th measurement is then modeled as

$$\mathbf{z}_i \in (N(R_i, \sigma_\rho^2), N(\Phi_i, \sigma_\phi^2)) \quad (11)$$

where $\mathbf{Z}_i = (R_i, \Phi_i)$ is the true i th measurement, $\sigma_\rho = 26$ mm, and $\sigma_\phi = 0.125^\circ$.

B. Validation Gates

In target tracking literature, the problem of data association has always been in focus. Using validation gates is a common way to handle the problem. A validation gate defines a region around some predicted value in which a measurement will be accepted as associated to the corresponding feature. We first try to filter out data points that are likely to be associated with the walls and then we extract parametric descriptions of these walls in the form of lines. Due to clutter, the walls might be very hard to find without prefiltering. The location of the gates will be functions of $\hat{\mathbf{x}}$ and \mathcal{M} . The size of the gates will depend on the quality of the sensor data, the method used to extract line parameters, the uncertainty in the environmental model, quantization

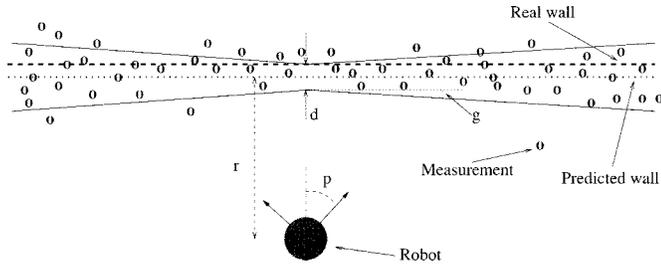


Fig. 9. Validation gate where the error in the predication of the wall and the size of the validation gate were scaled for illustration purposes.

step size, and P . The gates should open up, i.e., let more data through, when the uncertainty in the pose grows and vice versa. If the sensor data is very noisy, the gates have to be more open than if the sensor data is very “clean.”

Let the validation region be described by the sextuple $\mathcal{G} = (\hat{\rho}, \hat{\alpha}, \delta, \gamma, \varphi_1, \varphi_2)$. Here $\hat{\rho}$ is the predicted distance to the wall and $\hat{\alpha}$ is the predicted angle of normal to the line. These two entities define the pose of the gate. δ , the smallest width of the gate, and γ , the opening angle, define the size of the gate. φ_1 and φ_2 constitutes the *visibility constraint*. The visibility constraint amounts to not trying to detect a wall that is not in the field of view of the sensor (compare to Fig. 2). Fig. 9 shows a more detailed illustration of the parameters that define the location and size of the validation gates. Uncertainty in orientation will increase γ whereas uncertainty in position will demand higher δ .

Fig. 2 shows an example of how effective the validation gates are. The darker spots corresponds to scan points that have been associated with a map line, whereas the lighter ones are classified as outliers. As can be seen, we do not need to have a clear view of the walls, the only requirement is that we see points of a wall to make a total length of l_{\min} . In Fig. 2, the right and the lower wall was used for update, but not the upper one as enough of it was not seen.

C. Range Weighted Hough Transform (RWHT)

In order to track the walls reliably, we need to be able to extract them despite a large amount of clutter in the room. Just like Crowley *et al.* [5] and Wernersson *et al.* [16], we use the Hough transform. To be specific, we use the modified version introduced by Wernersson *et al.* [16] called the RWHT. In our algorithm, the RWHT is part of the filtering process that aims at providing a least squares-based line fitting algorithm with as clean range data as possible. To reduce computations, a local version of the RWHT is used with a limited Hough space, centered around the expected values $\hat{\rho}$ and $\hat{\alpha}$. That way, we do not have to perform all the calculations for lines that are not of interest to us.

The main purpose of this first filtering step is to handle situations where $\hat{\rho}$ and $\hat{\alpha}$ are only very rough estimates of the line parameters. That is, when the wall does not fall in the middle of the gate. Standard least squares algorithms are sensitive to outliers which calls for a narrow gate, whereas the RWHT allows the validation gates to be more open as it has proven to be very robust with respect to outliers.

D. Least Square Line Fitting Algorithm

In [17], a least squares algorithm is described for fitting range data to a line by minimizing the sum of squared perpendicular errors between the data points and the line. The distance d_i from the point (r_i, ϕ_i) to the line (ρ, α) is given by

$$d_i = r_i \cos(\phi_i - \alpha) - \rho \quad (12)$$

and the best fit (ρ, α) is thus given by

$$\arg \min_{(\rho, \alpha)} \sum_i (r_i \cos(\phi_i - \alpha) - \rho)^2. \quad (13)$$

The solution is (see [17])

$$\tan 2\alpha = \frac{p}{q} \quad (14)$$

$$p = \sum_i \sum_{j>i} r_i r_j \sin(\phi_i + \phi_j) + \sum_i (1 - N) r_i^2 \sin 2\phi_i \quad (15)$$

$$q = \sum_i \sum_{j>i} r_i r_j \cos(\phi_i + \phi_j) + \sum_i (1 - N) r_i^2 \cos 2\phi_i \quad (16)$$

$$\rho = \frac{1}{N} \sum_i r_i \cos(\phi_i - \alpha). \quad (17)$$

Using a first-order approximation and assuming independence between y_i and y_j for $i \neq j$, the covariance of the line parameters is given by

$$\begin{bmatrix} \sigma_{\rho}^2 & \sigma_{\rho\alpha}^2 \\ \sigma_{\rho\alpha}^2 & \sigma_{\alpha}^2 \end{bmatrix} = \begin{bmatrix} \frac{\partial \rho}{\partial \mathbf{r}} & \frac{\partial \rho}{\partial \boldsymbol{\phi}} \\ \frac{\partial \alpha}{\partial \mathbf{r}} & \frac{\partial \alpha}{\partial \boldsymbol{\phi}} \end{bmatrix} \begin{bmatrix} C_r & 0 \\ 0 & C_{\phi} \end{bmatrix} \begin{bmatrix} \frac{\partial \rho}{\partial \mathbf{r}} & \frac{\partial \rho}{\partial \boldsymbol{\phi}} \\ \frac{\partial \alpha}{\partial \mathbf{r}} & \frac{\partial \alpha}{\partial \boldsymbol{\phi}} \end{bmatrix}^T \quad (18)$$

where $C_r = \text{diag}(\sigma_{r_i}^2)$ and $C_{\phi} = \text{diag}(\sigma_{\phi_i}^2)$. The price that we pay for modeling the uncertainty of the individual measurement points in polar coordinates (as it should) is a higher computational complexity.

If we simplify matters by considering each data point to have the same Cartesian uncertainty, the uncertainty in the line parameters can be calculated much more efficiently according to (Deriche *et al.* [18])

$$\sigma_{\rho}^2 = A \quad (19)$$

$$\sigma_{\alpha}^2 = A + d^2 \frac{\sigma_{yy}^2 \cos^2 \chi + \sigma_{xx}^2 \sin^2 \chi - 2\sigma_{xy}^2 \sin \chi \cos \chi}{N} \quad (20)$$

where

$$\begin{aligned} \bar{x} &= \frac{1}{N} \sum x_i & \bar{y} &= \frac{1}{N} \sum y_i \\ a &= \sum (x_i - \bar{x})^2 & b &= 2 \sum (x_i - \bar{x})(y_i - \bar{y}) \\ c &= \sum (y_i - \bar{y})^2 & d &= \bar{y} \sin \chi + \bar{x} \cos \chi \\ \chi &= \left(\alpha + \frac{\pi}{2} \right) & A &= \frac{a\sigma_{yy}^2 - b\sigma_{xy}^2 + c\sigma_{xx}^2}{(a - c) + b^2}. \end{aligned}$$



Fig. 12. The corridor outside the living room and typical office in the CVAP building.

threshold between two rooms. To account for this increase in uncertainty, noise is injected into the system by increasing P , when the robot passes through a door. Along the track, different types of environments are encountered. Starting from the living room, the robot moves through the corridor to the nearby offices, which the robot moves into. These rooms are so small that the allowable movements of the robot are very limited (see Fig. 12). The offices are divided by cubicle dividers into two parts, leaving very limited sight of two of the walls. No problems have been found in these rooms during any of the runs though. The corridor at the ground floor of the laboratory is approximately 55 m long and the width is about 2.3 m (see Fig. 12). It is obvious that the problem in the corridor is going to be to maintain a good estimate of the position along the direction of the corridor. With all doors closed, the only sources of information about the position along the corridor are the short end walls. When moving far away ($> \approx 15\text{--}20$ m) from these short walls, they can no longer be used reliably, as too few points are accumulated. The detection of the short walls are made even more difficult if there is clutter in front of them. Because lines from neighboring rooms are used as well, a good estimate of the position along the corridor can still be maintained if some of the doors are open, providing a view of the walls inside. The standard deviation in the position estimate reached a maximum value of about 250 mm in the corridors.

Tracking is maintained for the duration of the run, the limiting factor in this experiment being the battery capacity and not the tracking algorithm.

C. Experiment 3

When comparing the performance of the pose tracking using the two different methods for calculating the least squares estimate of the lines, no detectable difference is found. Both methods have been run on the same data sets. Each

full update cycle (all lines) with the polar uncertainty model takes approximately 17 ms when run on a 400-MHz Pentium II. The corresponding number for the Cartesian uncertainty model is 4 ms. Given that the method based on a Cartesian assumption on the measurement points is less computationally intensive and that we ultimately want an algorithm that has low complexity, we see no reason not to use the simpler model for the uncertainty of the line parameters.

D. Experiment 4

In the last experiment, we tested how fast the update routine must be run to keep the tracking working. Simply put, the question does not have an answer if the environmental model is not specified. What is all comes down to is being able to solve the data association problem. In an environment with many large line-like structures that are parallel and close, the uncertainty must be kept small to distinguish between them. However, in most office environments, there are four dominating walls in each room. In such a case, the uncertainty can be allowed to grow larger.

Using the data from the 90-min run presented in Section VII-B, we tested running the algorithm at lower update rates with the conclusion that even with an update rate of as low as 0.05 Hz the robot tracks the pose. The uncertainty now becomes larger, but not large enough to cause erroneous data associations. Somewhere around 0.04 Hz, it breaks down for this data set, when passing from the living room out into the corridor. 0.04 Hz means one update every 25 s, during which the robot can have moved a significant distance. Updating at this rate does not reduce the uncertainty in the pose enough to maintain tracking.

E. Additional Comments

Besides the experiments described above, it is also important to mention that the algorithm has been tested successfully on four different platforms all having the laser scanner at different heights: Nomad200 (93 cm), Nomad SuperScout (52 cm), Nomadics XR4000 (49 cm), and a Pioneer2 (30 cm). Tests have also been made in environments different to the ones used above with many people walking around the robot.

The algorithm needs approximately 4 ms for each iteration (line extraction accounting for almost everything) on a PII 400 MHz. We have performed a comparison with a modified version [19] of what Fox *et al.* [8] call the Monte Carlo localization (MCL) method. Here a set of samples is used to approximate the probability density function (pdf) instead of the single Gaussian used in a Kalman filter. The complexity of the algorithm is linear in the number of samples in the set. To achieve similar tracking performance with a fixed sample set size, we need about 500 samples. The update of the sample set takes 24 ms, not including extraction of lines. Extracting lines is more time consuming when the approximate positions of the lines are unknown. The computational power needed is thus almost an order of magnitude higher than for the algorithm presented in this paper. The accuracy is not at all as good, the estimation error being in the order of 500 mm at times and far from being as accurate when the robot is moving slowly and has full-rank measurements (two nonparallel walls). Accuracy is not always

that important and the main advantage with the MCL method is that the sample set can represent multimodal pdfs. Therefore, it can handle situations where the robot due to one reason or the other becomes very uncertain of its position and data association is a problem. In fact, it is capable of handling the case of global uncertainty. The larger the uncertainty, the larger the sample set size and with it the computational demands. Using an adaptive sample set size allows for a smooth transition between global localization and pose tracking, which cannot be achieved with a single Gaussian. Multiple Gaussian can do it though [20].

VIII. DISCUSSION AND CONCLUSIONS

In this paper, we have presented a low complexity, highly robust and accurate pose tracking algorithm based on a minimalistic environmental model and a realistic model of the SICK laser scanner. The minimalistic environmental model provides robustness as only the large scale structures, such as the four dominating walls of a room, are captured. Such features are very likely to be robust over time and are relatively easy to extract due to their size, paving the way for low computational complexity. The low complexity of the algorithm is particularly important in an integrated system with limited computational resources.

Experiments show that the algorithm can handle a high density of clutter and is able to track the position for long periods of time. The limiting factor being the capacity of the batteries and not the algorithm.

Experiments also show that a simple model for the uncertainty of the line parameters is enough and that the algorithm can be run at very low frequency (below 0.1 Hz) and still maintains tracking. It is important to note that updating with low frequency increases the risk of losing track when something that is not captured by the odometric model occurs, such as slippage. A topic for future research is therefore to augment the pose tracker with means to detect when these events occur. This could be done by comparing relative motion information from a gyro and the odometry.

REFERENCES

- [1] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: System and Techniques*. Wellesley, MA: A. K. Peters, 1996.
- [2] J. L. Leonard, H. F. Durrant-Whyte, and I. J. Cox, "Dynamic map building for an autonomous mobile robot," *Int. J. Robot. Res.*, vol. 11, no. 4, pp. 286–298, 1992.
- [3] H. I. Christensen, N. O. S. Kirkeby, S. Kristensen, and L. F. Knudsen, "Model-driven vision for in-door navigation," *Robot. Autonom. Syst.*, vol. 12, pp. 199–207, 1994.
- [4] G. Hager and M. Mintz, "Sensor modeling and robust sensor data fusion," in *Proc. Int. Symp. Robotics Research*, 1990, pp. 69–74.
- [5] B. Schiele and J. L. Crowley, "A comparison of position estimation techniques using occupancy grids," in *Proc. Int. Conf. Robotics and Automation*, vol. 2, May 1994, pp. 1628–1634.
- [6] H. P. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proc. Int. Conf. Robotics and Automation*, 1985, pp. 116–121.
- [7] W. Burgard, D. Fox, D. Henning, and T. Schmidt, "Estimating the absolute position of a mobile robot using position probability grids," in *Proc. Nat. Conf. Artificial Intelligence*, 1996.

- [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*, 1999.
- [9] J. J. Leonard and H. F. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Trans. Robot. Automat.*, vol. 7, pp. 376–382, June 1991.
- [10] J. Forsberg, U. Larsson, P. Åhman, and Å. Wernersson, "Navigation in cluttered rooms using a range measuring laser and the hough transform," in *Proc. Int. Conf. Intelligence Autonomous Systems*, 1993, pp. 248–257.
- [11] W. D. Rencken, "Autonomous sonar navigation in indoor, unknown and unstructured environments," in *Proc. Int. Conf. Intelligence Autonomous Systems*, vol. 3, 1994, pp. 431–438.
- [12] J. L. Crowley, "World modeling and position estimation for a mobile robot using ultrasonic ranging," in *Proc. Int. Conf. Robotics and Automation*, 1989, pp. 674–680.
- [13] J. L. Crowley, F. Wallner, and B. Schiele, "Position estimation using principal components of range data," in *Proc. Int. Conf. Robotics and Automation*, Leuven, Belgium, May 1998, pp. 3121–3128.
- [14] J.-S. Gutmann, T. Weigel, and B. Nebel, "Fast, accurate and robust self-localization in polygonal environments," in *Proc. Int. Conf. Robotics and Automation*, 1999, pp. 1412–1419.
- [15] K. S. Chong and L. Kleeman, "Accurate odometry and error modeling for a mobile robot," *Intell. Robot. Res. Centre, Dept. Elect. Comput. Syst. Eng., Monash Univ., Clayton, Australia, Tech. Rep.*, 1996.
- [16] J. Forsberg, P. Åhman, and Å. Wernersson, "The hough transform inside the feedback loop of a mobile robot," in *Proc. Int. Conf. Robotics and Automation*, vol. 1, 1993, pp. 791–798.
- [17] K. O. Arras and R. Y. Siegwart, "Feature extraction and scene interpretation for map-based navigation and map building," in *Proc. SPIE, Mobile Robotics XII*, vol. 3210, 1997.
- [18] R. Deriche, R. Vaillant, and O. Faugeras, *From Noisy Edges Points to 3D Reconstruction of a Scene: A Robust Approach and Its Uncertainty Analysis*, Singapore: World Scientific, 1992, vol. 2, Series in Machine Perception and Artificial Intelligence, pp. 71–79.
- [19] P. Jensfelt, O. Wijk, D. Austin, and M. Andersson, "Experiments on augmenting condensation for mobile robot localization," in *Proc. Int. Conf. Robotics and Automation*, San Francisco, CA, May 2000, pp. 2518–2524.
- [20] P. Jensfelt and S. Kristensen, "Active global localization for a mobile robot using multiple hypothesis tracking," in *Proc. Int. Joint Conf. Artificial Intelligence*, Stockholm, Sweden, Aug. 1999, pp. 13–22.



Patric Jensfelt (M '??) received the M.Sc. degree from the School of Engineering Physics, Royal Institute of Technology, Stockholm, Sweden, in 1996. He is currently working toward the PhD degree with the Centre for Autonomous Systems (CAS).

His research interests include sensor fusion and in particular mobile robot localization.



Henrik I. Christensen received the M.Sc. and Ph.D. degrees from Aalborg University in 1987 and 1989, respectively.

He is a Chaired Professor of Computer Science at the Royal Institute of Technology, Stockholm, Sweden. He is also the Director of the Centre for Autonomous Systems. He has also held appointments at Aalborg University, University of Pennsylvania, and Oak Ridge National Laboratory. He has published more than 130 papers on robotics, vision, and integration. His primary research interest is system

integration for vision and mobile robotics.