Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Rao-Blackwellized Particle Smoothing for Occupancy-Grid Based SLAM Using Low-Cost Sensors [⋆]

Karl Berntorp [*] Jerker Nordh [*]

[*] *Department of Automatic Control, Lund University, SE–221 00 Lund, Sweden*
firstname.lastname@control.lth.se

Abstract: We approach the simultaneous localization and mapping problem by using an ultrasound sensor and wheel encoders on a mobile robot. The measurements are modeled to yield a conditionally linear model for all the map states. Moreover, we implement a Rao-Blackwellized particle smoother (RBPS) for jointly estimating the position of the robot and the map. The method is applied and successfully verified by experiments on a small Lego robot where ground truth was obtained by the use of a VICON real-time positioning system. The results show that the RBPS contributes with more robust estimates at the cost of computational complexity and memory usage.

1. INTRODUCTION

Having a robot simultaneously localizing itself and learning its map is commonly referred to as simultaneous localization and mapping (SLAM). The problem is often solved using odometry in combination with vision or range sensors. In mobile robotics it has been studied extensively over the last three decades. For surveys and tutorials of the SLAM problem and its different solutions up to recently, see for example [Thrun, 2002] or [Durrant-Whyte and Bailey, 2006].

At least since the early 1990s the approach to SLAM has been probabilistic. In [Smith et al., 1990], extended Kalman filtering (EKF) was used for state estimation. An issue with using Kalman filtering is that the nonlinearities that typically are present tend to lead to divergence of the state estimates. For example, the kinematics of a planar mobile robot is nonlinear in the heading angle, and the consequent linearizations that the EKF uses for estimating the odometry may lead to instability. To remedy this, particle filtering was introduced as a means to solve the SLAM problem. State-of-the-art particle filter algorithms when using high-resolution laser scanners are found in [Grisetti et al., 2005, 2007].

One way to describe the map is to use occupancy-grid mapping [Siciliano and Khatib, 2008]. The grids are considered to be either occupied or free, with some probability distribution associated with the grid. One usage of occupancy-grid maps is when utilizing range sensors, such as laser sensors or sonar sensors. Both types of sensors have noise and may occasionally give severe measurement errors. Since laser sensors have very high spatial resolution, thus giving a sharp probability distribution, they appear to be the most common solution, see [Hähnel et al., 2003], [Eliazar and Parr, 2003], [Grisetti et al., 2007]. In contrast, sonar sensors cover a cone in space. Because of the low spatial resolution in the tangential direction, it is impossible to determine from a single measurement whether a certain cell is occupied or not. Also, ultrasound sensors are sensitive to the angle of an object's surface relative to the sensor; that is, they have a small angle of incidence. This leads to that measurements of the same surface from slightly different angles may render different results. Obviously, this could potentially lead to estimation errors and must be handled by the algorithms.

In [Nordh and Berntorp, 2012] we performed SLAM using differential-driven mobile robots equipped with an ultrasound sensor. To handle the deficiencies with sonar sensors we extended the occupancy-grid concept by dividing every cell into subcells. Further, we developed a conditionally linear Gaussian state-space model for use in SLAM. In this paper we propose a novel measurement model aimed at sonar sensors, and extend the work in [Nordh and Berntorp, 2012] to include Rao-Blackwellized particle smoothing (RBPS) as a means for SLAM. Rao-Blackwellization takes advantage of a linear substructure in the model, which can be handled by a Kalman filter. Rao-Blackwellized particle filtering (RBPF) has been used for a number of years, see [Andrieu and Doucet, 2000], [Doucet et al., 2000], [Schön et al., 2005]. During the last years Rao-Blackwellized particle smoothers have gained interest, see [Särkkä et al., 2012], [Lindsten and Schön, 2011], where RBPS are developed for conditionally linear Gaussian models. We utilize the methods in [Lindsten and Schön, 2011] and provide an extension to also handle uniform noise for the class of differential-drive mobile robots.

[⋆] The authors are members of the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University. This work was supported by the Swedish Foundation for Strategic Research through the project ENGROSS. High accuracy reference measurements are provided through the use of the VICON real-time tracking system courtesy of the UAS Technologies Lab, Artificial Intelligence, and Integrated Computer Systems Division (AIICS) at the Department of Computer and Information Science (IDA), Linköping University, Sweden. See the URL http://www.ida.liu.se/divisions/aiics/aiicssite/index.en.shtm

Particle filters for SLAM tend to have a particle depletion problem, thus producing overconfident estimates of uncertainty and eliminating particles with low weights. This leads to that the number of particles used to perform loop closure decreases, yielding degenerate performance [Kwak et al., 2007]. Using particle smoothing could potentially eliminate this loop-closure problem since more information is available for map estimation and position localization.

We verify the method on a Lego Mindstorms mobile robot, see Fig. 1, equipped with a low-cost ultrasonic range finder. The Lego Mindstorms robot has low-performance motors, with severe backlash and highly uncertain encoder readings. Therefore, this setup represents a worst-case scenario.

1.1 Related Work

Using sonar sensors for SLAM has been studied before; an example is [Burgard et al., 1999], in which an offline expectation maximization algorithm was used for occupancy-grid mapping using 24 Polaroid sensors with 15 degrees opening angle. An early work is [Rencken, 1993], where the SLAM problem was solved in simulation using 24 ultrasonic sensors by estimating the errors introduced in the localization and mapping parts, respectively, and correcting for them using a modified Kalman filter approach. A third example is [Leonard et al., 1992], in which a feature-based approach with the aid of servo-mounted ultrasonic sensors was used. A more recent work is [Ahn et al., 2008], where ultrasonic sensors and a stereo camera is used in an EKF-SLAM setting. All of the previously mentioned work either use offline approaches and/or a vast number of sensors to perform SLAM. As such, their approaches are quite different from ours.

The state-of-the-art algorithms mentioned earlier, [Grisetti et al., 2005, 2007], are designed for use with laser scanners. Further, laser scanners, besides having high precision, are several orders of magnitude more expensive than the sensors we use. Therefore, a comparison with laser-based approaches would be unfair.

1.2 Outline

The structure of the rest of the paper is as follows: In Sec. 2 we give the preliminaries. Sec. 3 presents the state-space and measurement model, and Sec. 4 explains the forward filter used. In Sec. 5 we summarize the RBPS, derive how to handle uniform noise in the RBPS, and discuss implementation aspects. The algorithm is evaluated in Sec. 6. Finally, we conclude the paper in Sec. 7.

2. PRELIMINARIES

The conditional distribution density of the variable x at time index k conditioned on the variable y from time index i to time index k is denoted $p(x_k|y_{i:k})$. At each time step k the state can be partitioned into a nonlinear part ξ_k and a linear part z_k. The total state vector is a discrete-time Markov process, which obeys the transition density $p(\xi_{k+1}, z_{k+1}|\xi_k, z_k)$. The robot model is written as

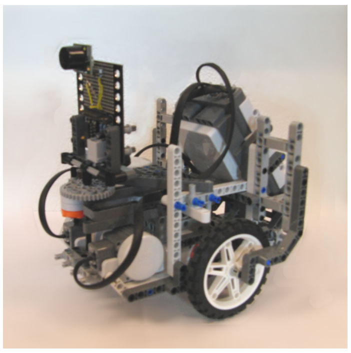

Fig. 1. The Lego Mindstorm differential-driven mobile robot used in the experiments. The ultrasound sensor is placed at the front end of the robot.

$$\xi_{k+1} = f(\xi_k, u_k, v_k, w_k), \tag{1}$$
$$z_{k+1} = A(\xi_k)z_k + e_{z,k}, \tag{2}$$
$$y_k^m(r_k, \xi_k) = C(r_k, \xi_k)z_k + e_{y^m,k}(r_k, \xi_k), \tag{3}$$

where $\xi \in \mathbb{R}^{7\text{x}1}$. The state vector $z \in \mathbb{R}^n$ contains the cells of the modified occupancy grid, where the size n depends on the map dimension, resolution, and the number of subcells used in every cell in the occupancy grid (for details see [Nordh and Berntorp, 2012]). The measurement function $C(\cdot)$ and measurement y^m are parametrized in the nonlinear state vector ξ and the ultrasound range measurement r. The inputs enter in the nonlinear states, and z is linear given ξ. The process noise v is independent uniform with zero mean, and w is white Gaussian with zero mean. Moreover, the process noise e_z and measurement noise e_{y^m} are assumed to be white Gaussian with zero mean.

3. MODELING

Here, we derive the robot kinematics and the conditionally linear map model. Moreover, we describe how the measurement equation is approximated as linear despite that the ultrasound sensor measurements are highly nonlinear.

3.1 State-Space Model

The robot used is assumed to be a differential-driven mobile robot, equipped with a sonar sensor. Since the robot moves in a plane, only three states are needed to describe the motion in continuous time. Using the position variables x and y, as well as the heading θ as state variables, the discretized kinematics (1) is, using a bilinear transformation, written as

$$\xi_{k+1} = f(\xi_k, u_k, v_k, w_k). \tag{4}$$

Here, $\xi_k = [x_k \ y_k \ \theta_k \ P_{k-1}^R \ P_{k-1}^L \ P_{k-2}^R \ P_{k-2}^L]^{\mathrm{T}}$ is the state vector, with $P_k^{R,L}$ being the right and left wheel encoder positions at time index k. The input u_k equals $u_k = [P_{k+1}^R \quad P_{k+1}^L]^{\mathrm{T}}$, and the wheel encoder noise vector is assumed uniformly distributed according to $v_k \sim U(-\alpha, \alpha)$. The process noise w_k only enters in θ with variance Q_w. After introducing

$$\bar{\theta}_k = \theta_k + \begin{bmatrix} \frac{1}{2l} & -\frac{1}{2l} \end{bmatrix} \begin{bmatrix} P_{k-2}^R \\ P_{k-2}^L \end{bmatrix} + \begin{bmatrix} \frac{1}{2l} & -\frac{1}{2l} \end{bmatrix} (u_k + v_k),$$

where l is the wheel axis length, the kinematics vector $f(\xi_k, u_k, v_k, w_k)$ becomes

$$f(\xi_k) = \begin{bmatrix} 1 & 0 & 0 & \frac{1}{4}a & \frac{1}{4}a & -\cos\theta_k & -\cos\theta_k \\ 0 & 1 & 0 & \frac{1}{4}b & \frac{1}{4}b & -\sin\theta_k & -\sin\theta_k \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \xi_k + \begin{bmatrix} \frac{1}{4}\cos\bar{\theta}_k & \frac{1}{4}\cos\bar{\theta}_k \\ \frac{1}{4}\sin\bar{\theta}_k & \frac{1}{4}\sin\bar{\theta}_k \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} (u_k + v_k) + \begin{bmatrix} 0 \\ 0 \\ \bar{\theta}_k + w_k \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

where $a = \cos\theta_k - \cos\bar{\theta}_k$, and $b = \sin\theta_k - \sin\bar{\theta}_k$. This model is affected both by the wheel encoder noise and the Gaussian distributed noise entering in θ. Wheel encoders typically have backlash uncertainties, which may be modeled as uniform noise. Moreover, the robot exhibits wheel slip, which provides the physical justification for w_k being Gaussian distributed. As we will discuss later the Gaussian noise can also be motivated with that it helps both the smoothing and filtering to perform better, by mitigating the issue of particle depletion.

The map is modeled as a slowly time-varying linear model; that is, every cell in (2) is modeled as

$$z_{k+1} = z_k + e_{z,k}, \tag{5}$$

where z contains the probability that a cell is occupied, stored in log-odds format. The process noise $e_{z,k}$ is a tuning parameter reflecting that the uncertainty of the cell grows when no new measurements arrive.

3.2 Measurement Model

Assume that an ultrasound measurement returns a distance to an object, and that the opening angle of the sensor is β degrees. The field of view is then a closed cone with aperture 2β. The cells that are inside the field of view can now be calculated, given that a position estimate exists. Assuming that a method exists for converting a single range measurement to an observation of the map states inside the field of view, the measurement equation would be linear. Further, the C-matrix in (3) would be sparse with a single 1 per row and with the same number of rows as the number of cells inside the field of view. A typical C matrix in (3) could be

$$y_k^m(r_k, \xi_k) = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \end{bmatrix} z_k + e_{y^m,k}(r_k, \xi_k). \tag{6}$$

The cells not inside the field of view at time index k do not generate any measurements at all for time index k.

To convert a range measurement to an observation of the map states, we set y_k^m to a high probability ($p = 0.99$) of occupancy for all the cells along the arc at the range of the distance measurement. All the cells closer than the measured distance are considered to be empty ($p = 0.01$). The additive noise is also parametrized by the distance so that when the detected object is far away the measurement is considered more noisy, thus suppressing the influence on the map. This reflects the high spatial uncertainty along the arc.

4. FORWARD FILTERING

The particle smoothing needs a weighted particle estimate as input. However, this estimate does not necessarily have to be generated by the same model as the one used for RBPS. In our case we use a slightly modified particle filtering approach.

For an RBPF in our setup, (1)–(3), the particle weights would be updated using the marginal density function of the linear states for the measurement. Instead we use a nonlinear function of the robot position and the map with a more physically intuitive formulation, which has proved effective in both simulations and on real data. Conditioned on the robot position all the cells that are inside the field of view are collected in a list and then sorted in ascending order of the range to the robot. The list is then traversed and the probability of each cell generating that measurement is calculated and weighted by the probability that all the earlier cells were empty, thus yielding

$$p(r_k) = \sum_{j=1}^{n} p(r_k|z_k^j) p_{\text{occ}}(z_k^j) \prod_{i=1}^{j-1} (1 - p_{\text{occ}}(z_k^i)), \tag{7}$$

where z_k^j is cell j at time index k. Each individual probability in (7) is evaluated against a flat probability density centered around the center point of the cell with width equal to the diagonal of a cell. Outside the flat region it falls off linearly over another cell width. See Fig. 2 for a visualization. This models the fact that a map with a certain resolution cannot differentiate measurements with higher precision than the resolution.

In a standard RBPF the weights of each of the N particles is updated by multiplying it with the new weight provided by the measurement. However, we have replaced this multiplication by a first order low-pass filter. The resulting algorithm is no longer a true RBPF, but from experience using both simulations and experimental data we have found that it works very well for our problem formulation.

5. RAO-BLACKWELLIZED PARTICLE SMOOTHING

In this section we only summarize the RBPS algorithm and provide our extension. The reader is referred to [Lindsten and Schön, 2011] for algorithm details.

5.1 Rao-Blackwellized Particle Smoother–Summary

The RBPS in [Lindsten and Schön, 2011] consists of three main steps:

(1) A backward particle smoother
(2) A forward Kalman filter
(3) A Rauch-Tung-Striebel (RTS) smoother

Backward Particle Smoother The backward pass starts with initializing $j = 1, \ldots, M$ backward trajectories at time T—that is, initializing $\{\xi_T^j, z_T^j\}$ using the filtered

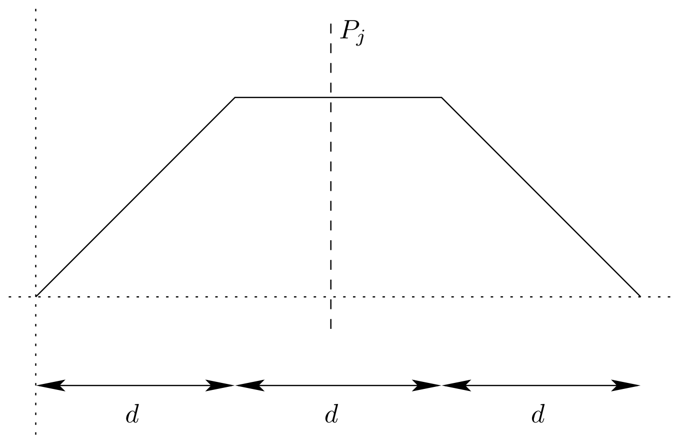

Fig. 2. The probability density function at time index k, $p(r_k|z_k^j)$, for a single measurement conditioned on cell j being the origin of that measurement. P_j is the center point of cell j, and d is the diagonal width of a cell

estimates at time T. Then, for the timespan of interest, rejection sampling is performed to find an index $I(j)$ corresponding to the forward filter particle that is to be appended to the j-th backward trajectory. This index is exploited to set $\xi_k^j = \xi_k^{I(j)}$ and to draw linear samples from the Gaussian density

$$\begin{aligned} p(z_k|\xi_{1:k}^{I(j)}, \xi_{k+1}, z_{k+1}, y_{1:k}^m) \propto p(\xi_{k+1}, z_{k+1}|\xi_{1:k}^{I(j)}, y_{1:k}^m) \\ \times p(z_k|\xi_{1:k}^{I(j)}, y_{1:k}^m,). \end{aligned} \tag{8}$$

Finally, the backward trajectory vector is appended, yielding $\{\xi_{k:T}^j, z_{k:T}^j\} = \{\xi_k^j, \xi_{k+1:T}^j, z_k^j, z_{k+1:T}^j\}$.

Kalman Filter After the backward smoothing step the nonlinear states are assumed fixed, thus yielding a linear model suitable for Kalman filtering [Anderson and Moore, 1979]. Hence, for each $j = 1, \ldots, M$ a Kalman filter runs for the whole timespan using (6).

RTS Smoother After the Kalman filter step both linear and nonlinear estimates are available. Then, an RTS smoothing (backward pass) step is performed, concluding the RBPS-algorithm. Note that steps 2 and 3 are needed to find continuous, unsampled, conditional smoothing densities.

5.2 RBPS with Uniform Noise

In the rejection sampling step in Sec. 5.1, [Lindsten and Schön, 2011] exploits that the forward density $p(\xi_{k+1}^j, z_{k+1}^j|\xi_{1:k}^i, y_{1:k}^m)$, where j is the trajectory index and i is the particle index, is Gaussian distributed. Also, calculation of the maximum of the distribution is needed. However, since we also have uniform noise in (4), some modifications are required:

From the factorization and Markov properties of (4) we have

$$\begin{aligned} p(\xi_{k+1}, z_{k+1}|\xi_{1:k}, y_{1:k}^m) &= p(\xi_{k+1}|\xi_{1:k})p(z_{k+1}|\xi_{1:k}, y_{1:k}^m) \\ &= p(\xi_{k+1}|\xi_k)p(z_{k+1}|z_k), \end{aligned} \tag{9}$$

where the input u_k is suppressed for reasons of notation. The second factor of (9) is the linear filtering density—that is, the density resulting from a Kalman filter. However, since the map is modeled as a slowly time-varying map, we have assumed it to be constant in the implementation. We can deduce the first factor in (9) as follows: Assume that we have the estimate ξ_{k+1}^*. Then the probability $p(\xi_{k+1} = \xi_{k+1}^*|\xi_k, u_k)$ can be reformulated as

$$p(\xi_{k+1} = \xi_{k+1}^*|\xi_k, u_k) = p(\xi_{k+1}^* = f(\xi_k, u_k, v_k, w_k)|\xi_k, u_k). \tag{10}$$

Due to the form of (4) we can solve $f(.)$ for v_k and w_k. Thus (10) transforms to

$$p((v_k,\, w_k) = g(\xi_{k+1}, \xi_k, u_k)|\xi_{k+1}, \xi_k, u_k). \tag{11}$$

Using the noise independence properties yields that (11) is equal to

$$p(v_k|g_1(\xi_{k+1}, \xi_k, u_k))p(w_k|g_2(\xi_{k+1}, \xi_k, u_k)). \tag{12}$$

To find the solution to (12), we start by forming the difference of (4) between two consecutive time steps (i.e., $\Delta x = x_{k+1} - x_k$). After reshuffling the equations for the translational coordinates we get

$$\cos\theta_{k+1} = \frac{4\Delta x - (P_k^R - P_{k-1}^R + P_k^L - P_{k-1}^L)\cos\theta_k}{P_{k+1}^R - P_k^R + P_{k+1}^L - P_k^L} \tag{13}$$

$$\sin\theta_{k+1} = \frac{4\Delta y - (P_k^R - P_{k-1}^R + P_k^L - P_{k-1}^L)\sin\theta_k}{P_{k+1}^R - P_k^R + P_{k+1}^L - P_k^L} \tag{14}$$

$$\Delta\theta = \frac{1}{2l}\left(P_{k+1}^R - P_{k-1}^R - (P_{k+1}^L - P_{k-1}^L)\right). \tag{15}$$

Utilizing the trigonometric identity on (13) and (14) gives that

$$(P_{k+1}^R + P_{k+1}^L - P_k^R - P_k^L)^2 = \gamma_1 \tag{16}$$

for some constant γ_1. Likewise, we get from (15) that

$$P_{k+1}^R - P_{k+1}^L = \gamma_2, \tag{17}$$

where $\gamma_2 = 2l\Delta\theta + P_{k-1}^R - P_{k-1}^L$. Moreover, by dividing (14) with (13) we can solve for the θ_{k+1} congruent to (x_{k+1}, y_{k+1}):

$$\theta_{k+1}^{1,2} = \operatorname{atan2}(d_1, d_2) + m\pi, \quad m = 0, 1, \tag{18}$$

where atan2$(\cdot,\cdot)$ is the four quadrant inverse tangent, and

$$\begin{aligned} d_1 &= 4\Delta x - (P_k^R - P_{k-1}^R + P_k^L - P_{k-1}^L)\cos\theta_k, \\ d_2 &= 4\Delta y - (P_k^R - P_{k-1}^R + P_k^L - P_{k-1}^L)\sin\theta_k. \end{aligned}$$

From (16) we see that there are two input vectors that give the same (x, y)-coordinates—that is,

$$P_{k+1}^R + P_{k+1}^L = P_k^R + P_k^L \pm \sqrt{\gamma_1}. \tag{19}$$

Also, from (15), (17), and (18), we find the two possible input vectors coincident to the two solutions of (19) to be

$$P_{k+1}^R - P_{k+1}^L = 2l\Delta\theta^{1,2} + P_{k-1}^R - P_{k-1}^L. \tag{20}$$

Using (19) and (20) we can calculate the (uniformly distributed) probability, $p(P_{k+1}^{R,L})$, of the two possible combinations of input vectors that yield the coordinates Δx, Δy, and $\Delta\theta^{1,2}$. The probability for the two possible solutions $\theta_{k+1}^{1,2}$ is Gaussian. Now (11) is formed by multiplying the Gaussian probability for $\theta_{k+1}^{1,2}$ with the two distributions for the wheel encoders and choosing the solution with the largest probability. Further, the maximum of $p(\xi_{k+1}, z_{k+1}|\xi_{1:k}, y_{1:k}^m, u_k)$ is found by setting $\theta_{k+1}^{1,2}$ to be in the middle of the distribution.

We now transform (8), which for our model yields

$$\begin{aligned} & p(z_k|\xi_{1:k+1}, z_{k+1}, y^m_{1:k}) \\ & \quad = p(z_k|\xi_{1:k}, z_{k+1}, y^m_{1:k}) \\ & \quad \propto p(z_{k+1}|z_k, \xi_{1:k}, y^m_{1:k}) p(z_k|\xi_{1:k}, y^m_{1:k}) \\ & \quad = p(z_{k+1}|z_k) p(z_k|\xi_{1:k}, y^m_{1:k}). \end{aligned} \tag{21}$$

Here, the first equality follows from that there is no new measurement at time index $k+1$, whereas the second and third steps follow from Bayes' rule and the Markov property, respectively. Now, (21) may be estimated using a Kalman filter.

Remark 1. We do not use an RBPF for finding the density (9), see Sec. 4. However, this is not a restriction since for the RBPS there is no assumption on how the density was produced.

Remark 2. The addition of the Gaussian state noise for θ can be motivated from that it can be used to model wheel slip. However, it also improves smoothing performance. Assume that we at time index k have the robot position estimate (x^0, y^0, θ^0), and that the estimate at time index $k+1$ is (x^1, y^1, θ^1). Then θ^1 is uniquely determined by the initial conditions and x^1 and y^1, since there are not enough degrees of freedom in the motion model, see Fig. 3 for an illustration. This means that the smoother will never switch between trajectories, caused by only having two degrees-of-freedom noise. This leads to that the smoother (and filter) will be unable to recover from spurious errors. However, if noise (i.e., an additional degree of freedom) is added to θ we remedy this, which is essential for the smoothing to give any impact, and for the forward filter to avoid particle depletion.

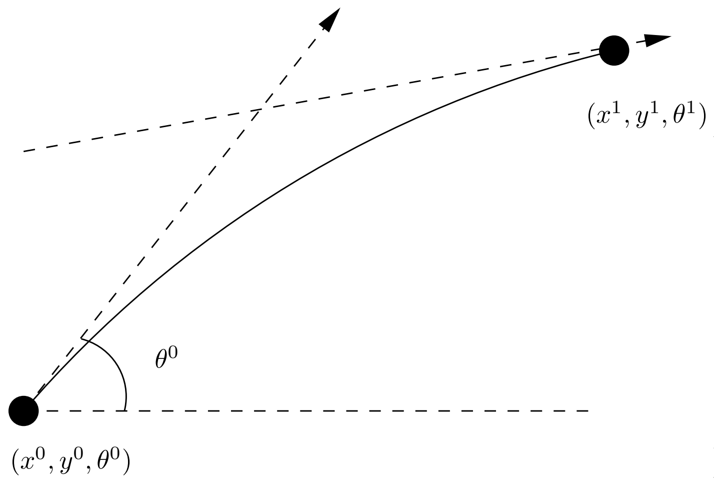

Fig. 3. The bilinear transformation (the arc) is a second order approximation of the robot motion. Thus, the end point is uniquely determined from the initial conditions and (x^1, y^1). This implies that when evaluating (9) for all j trajectories conditioned on particle i, the probabilities will equal zero when $j \neq i$. By adding noise for θ this uniqueness disappears, which implies that the smoother will be able to recover from errors caused by, for example, wheel slip.

5.3 Implementation Aspects

The computational demands when using smoothing are greater than for filtering. Not only the amount of computations needed increases, but there is also a need for storing the filtered estimate for each time instant over which the smoothing will be performed. For the problem described in this paper each particle contains an estimate of the entire map, thus it is of significant size. This means that for larger maps the memory requirements will be problematic unless some representation of the map which takes advantage of the similarities between particles at time instances is used. However, this is not an issue we investigated further in this work.

When implementing the RBPS for SLAM some parameter considerations have to be made. For example, the performance depends both on the time window used for the smoothing and the number of trajectories. Also, we will have to decide when to trigger the smoothing. Further, when the smoothing is finished it is not obvious from where we should reinitialize the forward filter. Since we are primarily interested in real-time estimation, it is not feasible to perform smoothing over the entire data set. We therefore use smoothing over a subset of the data. We then reinitialize the filtering at a point within the subset and restart from there. This provides the filtering with a "future" estimate of the map, which in theory should make it possible to perform better: Assume that the smoothing was triggered at time index k, and that we have a time window of length t. Thus, we suspend the forward filtering at time index k. Also assume that we initialize an additional forward filter halfway through the smoothing time window—that is, at time index $k - t/2$. Then we use the smoothed estimates at time index $k - t/2$ to initialize this additional forward filter, and executes it up to time index k. At time index k we may then attach these estimates and resume the original forward filter. Overlapping the filtering and smoothing in this way means that we can improve the future filtering with the smoothing. However, it increases the total amount of computations needed since we recompute the same time step several times.

The implementation as presented in this paper is available as open source software, see [Nordh and Berntorp, 2013b]. It is built on top of the open source framework pyParticleEst, see [Nordh and Berntorp, 2013a].

6. EXPERIMENTAL RESULTS

For the experiments we used a differential-driven mobile robot built using Lego Mindstorms, see Fig. 1. One of the aims of the present work is to use low-cost sensors, with rather poor performance. The sensor chosen was an ultrasonic range finder XL-Maxsonar EZ4 [MaxBotix Inc., 2012], with a resolution of 1 cm at 10 Hz. The maximum target range is 7.65 m, but because of voltage division in the building process the range was limited to approximately 3.8 m. Further, the maximum angle of incidence was measured to be approximately 10 deg. The sensor was mounted on a motor, enabling it to sweep back and forth. The motor used for this was of the same type as those used for driving the robot, which are part of the standard Lego Mindstorms toolkit. The backlash of the motors was estimated to roughly 5 deg. When it comes to precision in odometry, we believe that this setup represents a worst-case scenario.

The ground truth, only used for evaluation, was gathered using a VICON real-time positioning system, see Fig. 4,

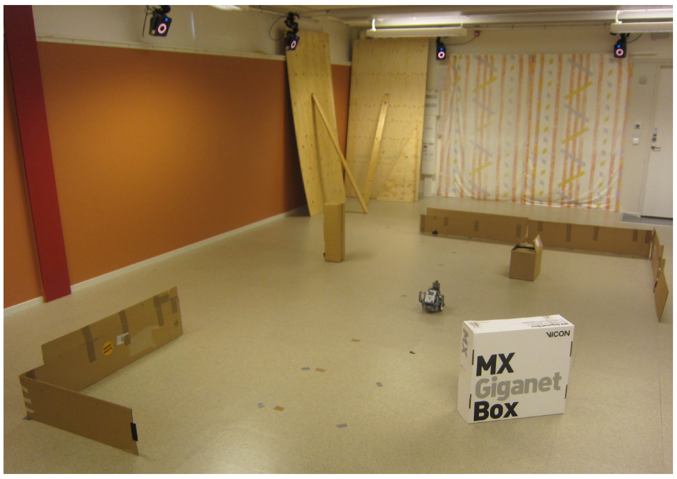

Fig. 4. The experimental setup. The cameras in the upper part are part of the VICON system.

installed at Linköping University, Sweden. The VICON system uses 10 infrared cameras and infrared lamps to track markers attached to the robot. The positioning precision provided by the system is about 1 mm.

The results were generated by effectuating the algorithm 40 times on the same data set. We set the number of backward trajectories to $M = 25$ and the number of forward particles to $N = 100$. The maximum smoothing length was set to 400. The reinitialization overlap was set to be half of the smoothing time window. Further, we chose to trigger the smoothing every fifth resample of the forward filter. These parameter choices were quite arbitrary, and most likely a better trade-off between complexity and performance can be found by tweaking the parameters.

We chose the mean of the norm of the position error at each time instance as performance measure. Figure 5 shows the results for smoothing-based SLAM, forward-filter SLAM, and dead reckoning. Included in the figure is the standard deviation for all three methods. It is clear that although the smoothing-based SLAM does not have much smaller mean error, it is more robust than only using the forward filter. The dead reckoning is deterministic since each execution was performed on the same data set. At about $t = 120$ the SLAM estimates seem to have converged to an error of about 0.4 m, which is of the same order of magnitude as the map resolution (0.2 m). The error in the odometry is clearly increasing with time, and gives substantially larger error than the SLAM algorithms in the position estimate for $t > 140$. Note that the SLAM algorithms start with no knowledge about its environment, and, hence, they do not outperform the odometry until sufficient map knowledge has been built up. The data set used was challenging in that during the last seconds the dead reckoning diverged quickly, with an angle error of about 180 deg and a position error of approximately 3 m. This was caused by severe wheel slip. The two SLAM algorithms managed to handle this in about 2/3 of the realizations.

Figure 5 is truncated in order to more clearly visualize the differences between the filtering and smoothing. In Fig. 6 we show results from a successful execution of the whole data set, where the smoothing-based SLAM

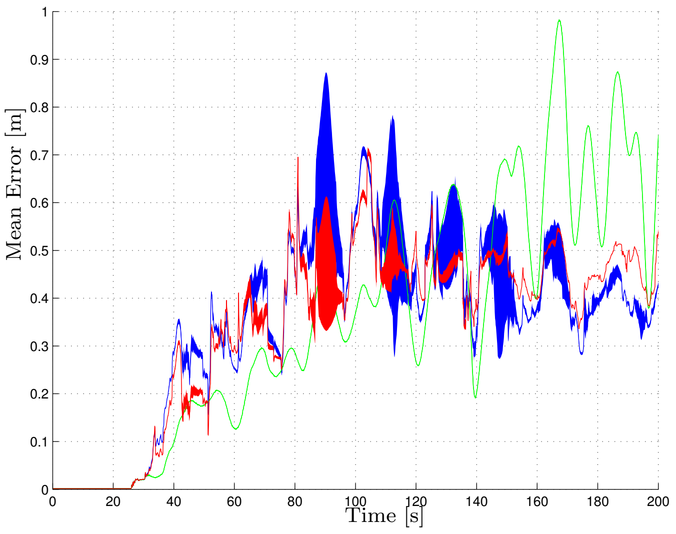

Fig. 5. Mean error and standard deviation for 40 realizations. The red curve shows the smoothing-based SLAM estimate, the blue curve shows the filtered SLAM estimate, and the green curve is the dead-reckoning estimate. The smoothing-based SLAM estimate is more consistent than both forward-filter SLAM and dead reckoning. Notice that initially neither the filtered nor the smoothed SLAM estimate outperforms the odometry, which is natural since initially there is no map estimate that can correct for the error in odometry. At around $t = 210$ a wheel slip occured that caused large odometry divergence. Therefore the data set is truncated at $t = 200$ in order to more clearly visualize the differences between the filtering and smoothing.

has a mean error that is roughly the same as the map resolution throughout the realization. Further, it manages to converge to the ground truth at the execution end time. Although the error of the forward-filter SLAM in Fig. 6 is from an unfortunate realization, it still shows that the smoothing-based SLAM is more robust and consistent. Results as inadequate as the forward-filter SLAM in Fig. 6 is not something we have observed when using the smoothing-based SLAM.

In Fig. 7 we visualize a map generated using the ground truth robot positions from the VICON system. In Fig. 8 we show the map obtained by the SLAM algorithm presented in this paper. For both cases each grid cell in the map actually consists of eight sub-cells corresponding to different directions in the environment according to the occupancy-grid extension mentioned in Sec. 1, see [Nordh and Berntorp, 2012]; what is visualized is the direction from which it is most likely to observe something, for each cell. In the maps presented the color blue represents areas that are unexplored by the sensor. The red color scale, starting from black, indicates the probability of a cell being occupied, starting from probability zero. The green color scale corresponds to the uncertainty (variance) of the probability estimate, with black implying large uncertainty and green corresponding to low uncertainty. Thus a yellow cell corresponds to that it is likely to be occupied, and there is low uncertainty associated with it. A green cell is believed to be empty with low uncertainty.

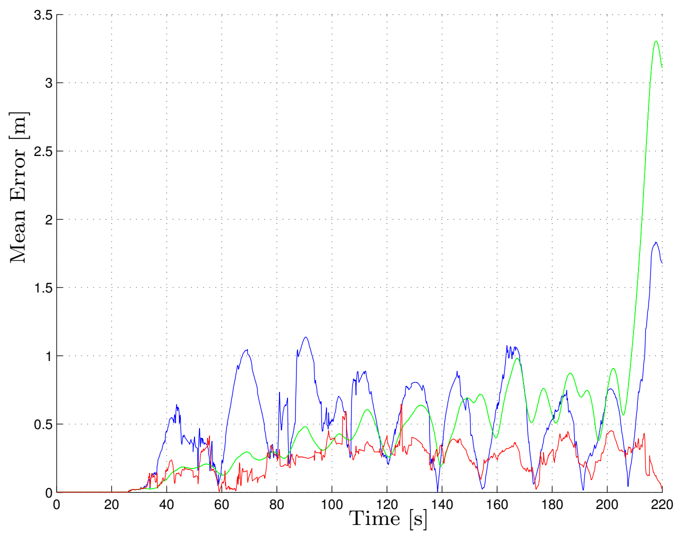

Fig. 6. Mean error for a single realization with same notation as in Fig. 5. The poor performance of the filtered estimates can be explained by that the map has converged with a bias compared to the real world, an issue which is largely corrected for by using smoothing instead. This data set is slightly longer than the one presented in Fig. 5 to clearly demonstrate the presented methods' abilities to handle severe odometry errors. Note the large error also for the filtering SLAM estimate (blue), something which the smoothing-based SLAM managed to correct for.

A red cell is believed to be occupied, but because of a lack of good measurements of that area the uncertainty is large. Similarly, black corresponds to areas believed to be empty, but with large uncertainty.

In Fig. 7 the position estimates for both the SLAM and dead reckoning are overlaid on the map, corresponding tho the last time step in Fig. 6. Also shown is the reference position from the VICON system. As seen the SLAM algorithm nearly coincides perfectly with the reference position at the end time, whereas the dead reckoning is several meters of, mainly due to the large wheel slip around $t = 210$. By visual inspection only, the map appears quite poor, and indeed it is of quite low resolution (each cell is 0.2 m wide) and rather noisy. But as can be seen from the SLAM position estimate it contains enough information to correct for the severe deficiencies in the dead reckoning. Looking for correspondences between the map and Fig. 4, we note that there is a concentration of yellow pixels roughly corresponding to the L-shaped wall in the upper right part of Fig. 4. There is also a blob of red pixels corresponding to the box in front of the L-shaped wall. The red arcs along the edges of the map are a result of the wide opening angle of the sensor; we simply have no other information other than that there somewhere along this arc exists an object. It is not possible to locate the object more precisely from the observations made. Note that the grids in the map are rarely yellow (occupied with low/zero uncertainty) or light green (empty with low/zero uncertainty), which is a major difference to regular occupancy grids.

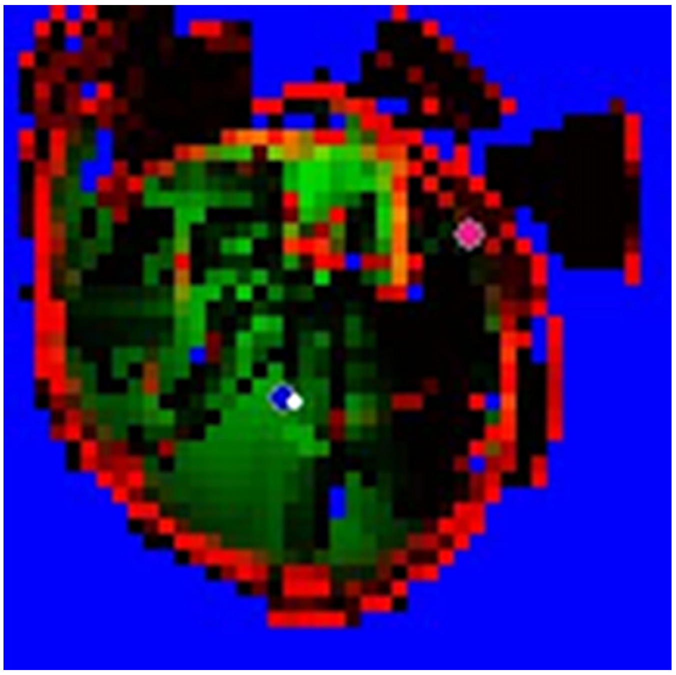

Fig. 7. The map corresponding to the last time step in Fig. 6, generated using the measurement model described in Sec. 3 but with the known positions from the VICON system. Since the position is known, this corresponds to the mapping part of the problem. Overlaid on this map is the position estimates; the white dot is the VICON reference, the blue dot close to the white is the SLAM estimate, and the pink dot is the dead reckoning. As can be seen the SLAM estimate has converged to the VICON ground-truth.

7. CONCLUSIONS AND FUTURE WORK

We presented a novel approach to the SLAM problem using an ultrasound sensor. A general method for differential-drive robots was developed, which uses a particle-filter inspired technique paired with a Rao-Blackwellized particle smoother. A more effective map model than the standard occupancy-grid formulation was used. The model incorporates the uncertainty of the measurements into the map. The experimental results show that the smoothing gives a substantial robustness improvement and increased predictability of the algorithm. Further, the results show that the algorithm is able to converge to the true position even for scenarios with extreme odometry uncertainty.

Possible future work is to incorporate initial estimates to improve the transient behavior of the method. Moreover, to implement path-finding algorithms that utilize the variance to explore the map is a natural extension of this paper.

REFERENCES

Sunghwan Ahn, Jinwoo Choi, Nakju Lett Doh, and Wan Kyun Chung. A practical approach for EKF-SLAM in an indoor environment: fusing ultrasonic sensors and stereo camera. *Auton. Robots*, 24:315–335, April 2008. ISSN 0929-5593.

Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Prentice Hall, Englewood Cliffs, NJ, 1979.

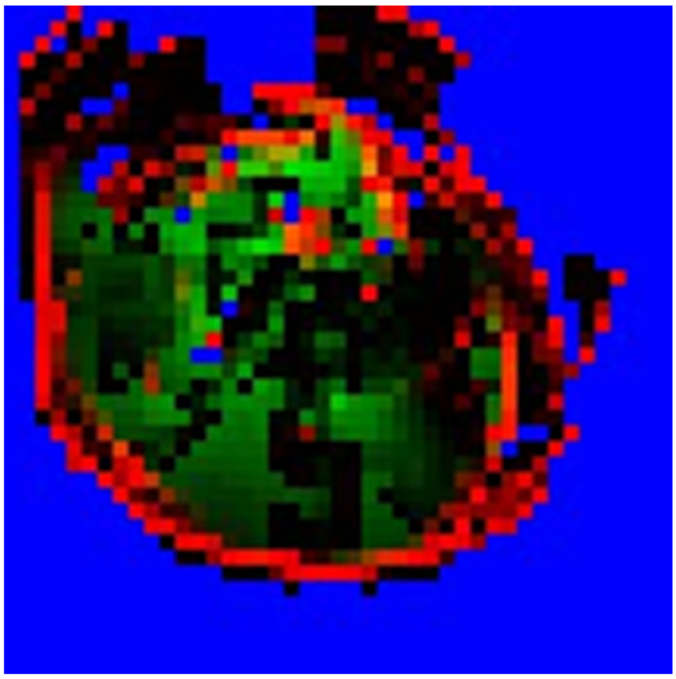

Fig. 8. The map corresponding to the last time step in Fig. 6, generated using the full SLAM algorithm as described in this article. Compared to the map in Fig. 7 this one is more noisy, and not all straight walls are preserved. But as can be seen from the accuracy of the position estimate (Figs. 5–7), this information is enough to compensate for inaccuracy in the odometry. Further, it is also able to compensate for the large wheel slip that occurred around time $t = 210$.

Christophe Andrieu and Arnaud Doucet. Particle filtering for partially observed Gaussian state space models. *J. R. Statist. Soc. B*, 64:827–836, 2000.

Wolfram Burgard, Dieter Fox, Hauke Jans, Christian Matenar, and Sebastian Thrun. Sonar-based mapping with mobile robots using EM. In *Proc. of the 16th Intl. Conf. on Machine Learning*, 1999.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3): 197–208, 2000.

Hugh Durrant-Whyte and Tim Bailey. Simultaneous localisation and mapping (SLAM): Part 2. *IEEE Robot. Autom. Mag.*, 13:108 –117, 2006.

Austin Eliazar and Ronald Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *Proc. 18th Int. Joint Conf. on Artificial Intell.*, pages 1135–1142. Morgan Kaufmann, 2003.

Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2443–2448, Barcelona, Spain, 2005.

Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Trans. Robot.*, 23: 34–46, 2007.

D Hähnel, W Burgard, D Fox, and S Thrun. A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 206–211, Las Vegas, NV, 2003.

Nosan Kwak, In K. Kim, Heon C. Lee, and Beom H. Lee. Analysis of resampling process for the particle depletion problem in FastSLAM. *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE Int. Symposium on*, pages 200–205, August 2007.

John J. Leonard, Hugh F. Durrant-Whyte, and Ingemar J. Cox. Dynamic map building for an autonomous mobile robot. *Int. J. Rob. Res.*, 11:286–298, August 1992. ISSN 0278-3649.

Fredrik Lindsten and Thomas Schön. Rao-Blackwellised particle smoothers for mixed linear/nonlinear state-space models. Tech. Report LiTH-ISY-R-2018, Div. of Automatic Control, Linköping Univ., Sweden, May 2011.

MaxBotix Inc. XL-MaxSonar-EZ4 Datasheet, 2012. URL: http://www.maxbotix.com.

Jerker Nordh and Karl Berntorp. Extending the occupancy grid concept for low-cost sensor based SLAM. In *Proc. of the 10th Int. IFAC Symp. on Robot Control*, volume 10, pages 151–156, Dubrovnik, Croatia, September 2012.

Jerker Nordh and Karl Berntorp. pyParticleEst - a Python framework for particle based estimation. Tech. Report ISRN LUTFD2/TFRT--7628--SE, Department of Automatic Control, Lund Univ., Sweden, March 2013a.

Jerker Nordh and Karl Berntorp. NXT SLAM, 2013b. URL: www.control.lth.se/JerkerNordh/nxt_slam.html.

W.D. Rencken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *Proc. of the 1993 IEEE/RSJ Int. Conf. on IROS '93*, volume 3, pages 2192–2197 vol.3, jul 1993.

S. Särkkä, P. Bunch, and S. J. Godsill. A backward-simulation based Rao-Blackwellized particle smoother for conditionally linear Gaussian models. In *Proc. of SYSID 2012*, volume 16, pages 506–511, Brussels, Belgium, 2012.

T. Schön, F. Gustafsson, and P.-J. Nordlund. Marginalized particle filters for mixed linear/nonlinear state-space models. *IEEE Trans. Signal Process.*, 53(7):2279–2289, July 2005.

Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer, Berlin, Heidelberg, 2008. ISBN 978-3-540-23957-4.

R. Smith, M. Self, and P. Cheeseman. *Estimating uncertain spatial relationships in robotics*, pages 167–193. Springer-Verlag New York, Inc., New York, NY, 1990. ISBN 0-387-97240-4.

S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.