Efficient Estimation of Expected Distributions for Mobile Robot Navigation

Maximilian Beinhofer and Wolfram Burgard

Abstract— **In real-world applications, motion and sensor observations of mobile robots are typically subject to noise and errors. A common way to deal with this problem is to represent the states of a robot at different points in time with probability distributions. Being able to calculate the expected distributions of the states of the robot even before the robot starts operation is useful for many robotics problems including path planning and optimizing the configuration in which to mount sensors on a mobile robot. When searching for optimal solutions to these problems, one typically wants to compute such expected distributions for a large number of candidate solutions. Therefore, it is highly important to have an efficient way of calculating the expected distributions. In this paper, we present a novel approach for efficiently estimating the expected distributions of the states of a mobile robot that uses a linear-quadratic regulator controller designed to guide the robot along pre-calculated trajectories. We exploit the structure of the stochastic dependencies in the navigation framework for deriving a recursive procedure to calculate the expected distributions. Compared to the state of the art, this procedure reduces the dimensionality of the occurring matrix multiplications by half. In extensive experiments, we show that the reduced dimensionality leads to a considerable reduction of the computation time without loss of information.**

I. INTRODUCTION

Mobile robots operating in the real world typically have to deal with errors in motion execution and sensor observations. A standard approach to account for these errors is to describe the evolution of the state of the robot during operation by a series of probability distributions [12]. When applying this approach to mobile robot localization, the goal is to estimate the posterior probability distributions of the state of the robot, i.e., the distributions that are conditioned on the already executed controls and observations. In contrast to that, we are interested in estimating expected distributions of the state of the robot during operation that depend only on the information about the robot and its task available before the robot starts operating, and not on the concrete controls and observations.

Being able to estimate these expected distributions is useful in several robotics problems like path planning [3, 11, 14], optimizing the configuration of sensors on a mobile robot [9], or landmark placement [2]. In landmark placement, for example, expected distributions can be used to find positions for artificial landmarks in the environment of a mobile robot that minimize the uncertainty about its deviation from its desired trajectory. When searching for the optimum, these kinds of approaches typically need to evaluate a large number of candidate solutions (landmark sets, candidate paths, sensor configurations), which makes a short calculation time for the expected distributions especially important.

In this paper, we present a novel method for efficiently estimating the expected distributions of the states of a mobile robot that is controlled by a linear-quadratic regulator (LQR) controller [5]. Our approach linearizes the model of the whole navigation cycle, including control, motion, and observation, and recursively calculates the expected distributions in the linearized system.

The contribution of this paper is twofold: first, we present an approach that considerably reduces the runtime of the computation of expected distributions compared to the state of the art. Second, the derivation of our approach yields theoretical insights into the considered widely-used [2, 3, 10, 14] linearized system for mobile robot navigation. It builds on the fact that before the robot starts operating, the posterior localization estimate $\boldsymbol{\mu}_t$ of the robot can be considered as a random variable, which is highly correlated to the state of the robot $\mathbf{x}_t$. We show that this correlation has a structure that allows us to decouple the calculation of the covariances of $\boldsymbol{\mu}_t$ and $\mathbf{x}_t$. With this, our approach can recursively update the distributions of $\boldsymbol{\mu}_t$ and $\mathbf{x}_t$ individually, whereas the state of the art [3] recursively updates their joint distribution. Therefore, compared to [3], our approach reduces the dimensionality of the occurring matrix multiplications by half, which results in a substantial reduction of the runtime of the computations. In extensive experiments, we show that our approach significantly outperforms other approaches in terms of runtime, while still producing exactly the same results.

II. RELATED WORK

There are several ways to estimate expected distributions for dynamic systems: Possibly the most generally applicable, but also computationally most demanding method is Monte Carlo simulation. Roy *et al.* [11], for example, use Monte Carlo simulation to estimate the expected entropy of the robot state in their coastal navigation framework. Another method is to use the posterior distributions of the most likely run of the robot as approximations for the expected distributions. Vitus and Tomlin [13] use this method for sensor placement. For a given desired robot trajectory, they aim at placing sensors at a set of locations in the environment that optimizes the navigation performance of the robot. Mastrogiovanni *et al.* [9] also use the posterior distributions to estimate the pose uncertainty of a robot before operation. They, however,

This work has partly been supported by the German Research Foundation (DFG) within the Research Training Group 1103 and under contract number SFB/TR 8. The authors are with the Department of Computer Science, University of Freiburg, Germany. beinhofe@informatik.uni-freiburg.de

use these distributions to find the optimal configuration for mounting a laser scanner on a mobile robot.

To our knowledge, van den Berg *et al.* [3] were the first to introduce a recursive calculation scheme for expected distributions in dynamic systems. They call the expected distributions *a priori distributions*, and used them for collision-free path planning. Later, they applied their approach to needle steering for surgical robots [4]. Since then, their calculation scheme has been applied to different kinds of applications: Vitus and Tomlin [14] used it for chance constrained optimal control, Patil *et al.* [10] applied it to motion planning in deformable environments, and Beinhofer *et al.* [2] employed it for landmark placement.

The method presented in this paper calculates expected distributions considerably faster than the one by van den Berg *et al.* and could therefore benefit all of the above approaches.

III. THE ROBOTIC SYSTEM

We consider the problem of estimating the expected distributions of the states $\mathbf{x}_t$ of a mobile robot traveling along a predefined trajectory $\mathcal{T}$. Hereby, the trajectory $\mathcal{T} = (\mathbf{x}^\star_{0:T},\, \mathbf{u}^\star_{1:T})$ is a time-discrete sequence specifying the desired robot state $\mathbf{x}^\star_t$ and control command $\mathbf{u}^\star_t$ at each time step t. The actual robot state $\mathbf{x}_t$ changes over time according to the stochastic motion model

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{v}_t)\,, \tag{1}$$

where $\mathbf{u}_t$ is the actual control command executed at time t and $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, Q_t)$ is the motion noise, which we assume to be Gaussian distributed. Due to the stochastic nature of the motion model, the actual robot states $\mathbf{x}_t$ differ from the desired states $\mathbf{x}^\star_t$. To reduce this difference, the robot needs to execute control commands $\mathbf{u}_t$ that differ from the desired control commands $\mathbf{u}^\star_t$. We assume that the robot uses an LQR controller [5] to select the control commands. At each time step t, the LQR controller executes the control $\mathbf{u}_t$ that minimizes the expected quadratic error term

$$\mathbb{E}\Big[\sum_{\ell=t}^{T}((\mathbf{x}_\ell - \mathbf{x}^\star_\ell)^T C(\mathbf{x}_\ell - \mathbf{x}^\star_\ell) + (\mathbf{u}_\ell - \mathbf{u}^\star_\ell)^T D(\mathbf{u}_\ell - \mathbf{u}^\star_\ell))\Big], \tag{2}$$

where C and D are positive definite weight matrices.

For localization, the robot has a map of its environment and a sensor that takes noisy observations $\mathbf{z}_t$ of the surroundings of the robot according to a sensor model

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{w}_t)\,, \tag{3}$$

where $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, R_t)$ is the sensor noise.

A. Expected Distributions

To estimate the states of a mobile robot during operation, one typically applies some kind of filtering framework [12] to estimate the posterior probability distributions $p(\mathbf{x}_t \mid \mathbf{u}_{1:t}, \mathbf{z}_{1:t})$ of the states of the robot, which are conditioned on the already executed controls and observations. In contrast to that, we are interested in estimating the expected distributions of the state $\mathbf{x}_t$ of the robot even before it starts operating. Before operation, the concrete controls $\mathbf{u}_{1:t}$ and observations $\mathbf{z}_{1:t}$ are not yet known. Therefore, the expected distributions $p(\mathbf{x}_t)$ depend only on the trajectory $\mathcal{T}$, the motion model f, and the sensor model h. Using the law of total probability, we can relate the expected distributions $p(\mathbf{x}_t)$ to the posterior distributions: $p(\mathbf{x}_t) =$

$$\int\!\!\int p(\mathbf{x}_t \mid \mathbf{u}_{1:t}, \mathbf{z}_{1:t})\, p(\mathbf{u}_{1:t}, \mathbf{z}_{1:t})\, \mathrm{d}\mathbf{u}_{1:\mathrm{t}}\, \mathrm{d}\mathbf{z}_{1:\mathrm{t}}\,. \tag{4}$$

In general, $p(\mathbf{x}_t)$ cannot be estimated in closed form, so one solution that is often applied is to approximate the high-dimensional integral defined in (4) via Monte Carlo simulation. Monte Carlo simulation can deal with arbitrary robot models, but is computationally demanding.

In contrast to that, we efficiently estimate $p(\mathbf{x}_t)$ by linearizing the whole navigation system, consisting of observation, localization, control, and motion, resulting in a Gaussian expected distribution that can be calculated efficiently via standard matrix manipulations.

B. The Linearized System

For linearizing the motion model (1) and the sensor model (3), it is convenient to consider the deviations of the states, controls, and observations, from their desired values instead of the absolute values themselves. Therefore, we define

$$\Delta\mathbf{x}_t := \mathbf{x}_t - \mathbf{x}^\star_t,\ \Delta\mathbf{u}_t := \mathbf{u}_t - \mathbf{u}^\star_t,\ \Delta\mathbf{z}_t := \mathbf{z}_t - h(\mathbf{x}^\star_t, \mathbf{0}).$$

For the linearization procedure, we follow the approach by van den Berg *et al.* [3] and use first-order Taylor expansions around the desired trajectory $(\mathbf{x}^\star_{0:T},\, \mathbf{u}^\star_{1:T})$, leading to the approximate identities

$$\Delta\mathbf{x}_t \approx A_t\Delta\mathbf{x}_{t-1} + B_t\Delta\mathbf{u}_t + V_t\,\mathbf{v}_t\,, \tag{5}$$

$$\Delta\mathbf{z}_t \approx H_t\Delta\mathbf{x}_t + W_t\,\mathbf{w}_t\,. \tag{6}$$

with the Jacobians

$$\begin{aligned} A_t = \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}^\star_{t-1}, \mathbf{u}^\star_t, \mathbf{0}), \quad B_t = \frac{\partial f}{\partial \mathbf{u}}(\mathbf{x}^\star_{t-1}, \mathbf{u}^\star_t, \mathbf{0}),\\ V_t = \frac{\partial f}{\partial \mathbf{v}}(\mathbf{x}^\star_{t-1}, \mathbf{u}^\star_t, \mathbf{0}),\\ H_t = \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^\star_t, \mathbf{0}), \quad W_t = \frac{\partial h}{\partial \mathbf{w}}(\mathbf{x}^\star_t, \mathbf{0})\,. \end{aligned} \tag{7}$$

In this linearized system, the Gaussian posterior distribution $p(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t}) \sim \mathcal{N}(\Delta\boldsymbol{\mu}_t, \Sigma_t)$ of the deviation from the trajectory can be computed recursively using a Kalman Filter [1]. The Kalman Filter propagates a given initial Gaussian distribution $p(\Delta\mathbf{x}_0) \sim \mathcal{N}(\Delta\boldsymbol{\mu}_0, \Sigma_0)$ according to the update scheme

$$\overline{\Delta\boldsymbol{\mu}}_t = A_t\Delta\boldsymbol{\mu}_{t-1} + B_t\Delta\mathbf{u}_t \tag{8}$$

$$\overline{\Sigma}_t = A_t\Sigma_{t-1}A_t^T + V_tQ_tV_t^T \tag{9}$$

$$K_t = \overline{\Sigma}_tH_t^T(H_t\overline{\Sigma}_tH_t^T + W_tR_tW_t^T)^{-1} \tag{10}$$

$$\Delta\boldsymbol{\mu}_t = \overline{\Delta\boldsymbol{\mu}}_t + K_t(\Delta\mathbf{z}_t - H_t\overline{\Delta\boldsymbol{\mu}}_t) \tag{11}$$

$$\Sigma_t = (I - K_tH_t)\overline{\Sigma}_t\,. \tag{12}$$

Note that the covariance Σ_t and the Kalman gain K_t depend, via the Jacobians, on $\mathbf{x}^\star_{0:t}$ and $\mathbf{u}^\star_{1:t}$ but not on the

actual values of $\mathbf{u}_{1:t}$ and $\mathbf{z}_{1:t}$. Therefore they can be calculated before the robot starts operation.

Applied on the mean $\Delta\boldsymbol{\mu}_{t-1}$ in the Kalman Filter, the LQR controller selects motion commands $\mathbf{u}_t$ according to

$$\Delta\mathbf{u}_t = L_t \Delta\boldsymbol{\mu}_{t-1}\,, \tag{13}$$

where L_t is the feedback matrix that minimizes the quadratic error defined in (2). L_t depends only on the Jacobians in (7) and the weight matrices C and D and therefore can be calculated before the robot starts operating. See [5] for a derivation of the explicit calculation formula for L_t.

Summing up, we express the whole navigation cycle, which consists of executing a motion command, making an observation, localizing, and selecting the next motion command depending on the localization, by linear functions.

IV. EXPECTED DISTRIBUTIONS IN THE LINEARIZED SYSTEM

In this section, we present our novel method for efficiently calculating expected distributions via recursion for the above-described linearized robotic system. The proofs of all theorems in this section can be found in the appendix.

A. The Efficient Calculation Scheme

For the derivation of our efficient calculation scheme, the mean $\Delta\boldsymbol{\mu}_t$ of the posterior distribution plays a key role. Before the robot starts operating, i.e., before making any observations and selecting any motion commands, $\Delta\boldsymbol{\mu}_t$ can be considered as a random variable, which deterministically depends on $\mathbf{u}_{1:t}$ and $\mathbf{z}_{1:t}$.

Lemma 1. *Assuming that $\Delta\mathbf{x}_0$ is zero-mean Gaussian distributed, the expected distributions of $\Delta\mathbf{x}_t$ and of $\Delta\boldsymbol{\mu}_t$ are Gaussians with zero mean for all time steps t, i.e.,*

$$\forall t \in [0,T]:\ p(\Delta\mathbf{x}_t) \sim \mathcal{N}(\mathbf{0}, S_t)\,, \quad p(\Delta\boldsymbol{\mu}_t) \sim \mathcal{N}(\mathbf{0}, M_t)\,.$$

Up until here, we loosely followed the approach by van den Berg *et al.* [3]. However, in the following, we apply techniques different from theirs to derive a more efficient way for computing the covariances S_t of the expected distributions. The main building block for this is the following theorem.

Theorem 1. *The covariance S_t of the expected distribution of $\Delta\mathbf{x}_t$ is the sum of the posterior covariance Σ_t of $\Delta\mathbf{x}_t$ and the covariance M_t of the expected distribution of the mean $\Delta\boldsymbol{\mu}_t$ in the Kalman Filter:*

$$\forall t \in [0,T]: \quad S_t = \Sigma_t + M_t\,. \tag{14}$$

This also has strong implications on the cross-covariance of $\Delta\mathbf{x}_t$ and $\Delta\boldsymbol{\mu}_t$:

Corollary 1. $\mathit{Cov}(\Delta\mathbf{x}_t, \Delta\boldsymbol{\mu}_t) = \mathit{Cov}(\Delta\boldsymbol{\mu}_t)\ (= M_t)$.

For being able to use Theorem 1 to calculate S_t, we need to be able to calculate both Σ_t and M_t before the robot starts operating. How to calculate Σ_t in the linearized system is already clear from Equations (9, 10, 12). Note that this calculation only uses values that are available before the robot starts operating. In the following, we derive an efficient recursive update formula for the covariance M_t of $\Delta\boldsymbol{\mu}_t$. To do so, we first consider the covariance of the difference between $\Delta\mathbf{x}_t$ and the mean $\overline{\Delta\boldsymbol{\mu}_t}$ of the posterior distribution $p(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t-1})$ before the integration of observation $\Delta\mathbf{z}_t$.

Algorithm 1 Recursive Calculation of $S_{0:T}$

Input: $S_0 (= \Sigma_0)$, $M_0 = 0$
Output: $S_{0:T}$

```
for t = 1 to T do
    M̄_t ← (A_t + B_t L_{t-1}) M_{t-1} (A_t^T + L_{t-1}^T B_t^T)
    Σ̄_t ← A_t Σ_{t-1} A_t^T + V_t Q_t V_t^T
    K_t ← Σ̄_t H_t^T (H_t Σ̄_t H_t^T + W_t R_t W_t^T)^{-1}
    M_t ← M̄_t + K_t H_t Σ̄_t
    Σ_t ← (I − K_t H_t) Σ̄_t
    S_t ← Σ_t + M_t
end for
return S_{0:T}
```

Lemma 2.

$$\mathit{Cov}(\Delta\mathbf{x}_t - \overline{\Delta\boldsymbol{\mu}_t}) = \mathit{Cov}(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t-1}) = \overline{\Sigma}_t.$$

With this, we can derive an efficient recursive update formula for the covariance M_t of $\Delta\boldsymbol{\mu}_t$:

Lemma 3.

$$M_0 = \mathbf{0}\,, \tag{15}$$
$$\overline{M}_t = (A_t + B_t L_{t-1}) M_{t-1} (A_t^T + L_{t-1}^T B_t^T)\,, \tag{16}$$
$$M_t = \overline{M}_t + K_t H_t \overline{\Sigma}_t\,, \tag{17}$$

where $M_t = \mathit{Cov}(\Delta\boldsymbol{\mu}_t)$ and $\overline{M}_t = \mathit{Cov}(\overline{\Delta\boldsymbol{\mu}_t})$.

Summing up, we know from Lemma 1 that the expected distribution of $\Delta\mathbf{x}_t$ is $\mathcal{N}(\mathbf{0}, S_t)$, and therefore that the expected distribution of $\mathbf{x}_t$ is $\mathcal{N}(\mathbf{x}_t^\star, S_t)$. Hence, the mean of the distribution is already known from the desired trajectory, and we only need to calculate the covariance S_t. We calculate S_t using the fact that $S_t = \Sigma_t + M_t$, as shown in Theorem 1. We know the recursive calculation schemes for Σ_t and for M_t from the equations in Section III-B and from Lemma 3, respectively. Both only depend on terms that can be computed before the robot starts operating. Algorithm 1 states the complete recursive calculation method for the covariances of the expected distributions.

B. Comparison to the State of the Art

To our knowledge, the first approach for calculating expected distributions in dynamic systems recursively was derived by van den Berg *et al.* [3]. In the same linear system that we described above, they consider the expected joint distribution of $\Delta\mathbf{x}_t$ and $\Delta\boldsymbol{\mu}_t$, which is a Gaussian

$$p\left([\Delta\mathbf{x}_t\ \ \Delta\boldsymbol{\mu}_t]^T\right) \sim \mathcal{N}\left([\mathbf{0}\ \ \mathbf{0}]^T, J_t\right)\,, \tag{18}$$

with

$$J_t = \begin{bmatrix} S_t & \mathrm{Cov}(\mathbf{x}_t, \boldsymbol{\mu}_t) \\ \mathrm{Cov}(\mathbf{x}_t, \boldsymbol{\mu}_t)^T & M_t \end{bmatrix}\,. \tag{19}$$

They do not decouple the calculation of S_t and M_t as we do through Theorem 1, but they derive the following recursive update scheme for the joint covariance J_t:

$$J_0 = \begin{bmatrix} \Sigma_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{20}$$

$$J_t = F_t J_{t-1} F_t^T + G_t \begin{bmatrix} Q_t & \mathbf{0} \\ \mathbf{0} & R_t \end{bmatrix} G_t^T, \tag{21}$$

with

$$F_t = \begin{bmatrix} A_t & B_t L_{t-1} \\ K_t H_t A_t & A_t + B_t L_{t-1} - K_t H_t A_t \end{bmatrix}, \tag{22}$$

$$G_t = \begin{bmatrix} V_t & 0 \\ K_t H_t V_t & K_t W_t \end{bmatrix}. \tag{23}$$

They then can extract S_t as the upper left block of J_t. As can be seen from the equations, their approach requires multiplications of matrices of dimension $(2\dim(\mathbf{x}_t) \times 2\dim(\mathbf{x}_t))$, while our approach multiplies only matrices of dimension $(\dim(\mathbf{x}_t) \times \dim(\mathbf{x}_t))$. Besides this, their approach needs exactly the same calculations as ours, because they also need to calculate Σ_{t-1} in order to compute K_t. So all other computations being equal, the final recursive calculation of S_t in our approach is 2^3 times faster than the one by van den Berg *et al.* (if the standard matrix multiplication algorithm for $n \times n$-matrices with runtime n^3 is applied). In the next section, we present a detailed comparison of the runtimes in practice.

V. EXPERIMENTAL EVALUATION

We evaluated our approach in extensive experiments with a differential drive robot motion model and a sensor model for measurements consisting of the distance and the relative angle between the robot and a set of uniquely identifiable landmarks. In the experiments, we describe the state $\mathbf{x}_t$ of the robot by its pose $[x_t, y_t, \theta_t]$ in the 2d-plane. We use a discretization of 10 Hz for the time steps, resulting in trajectories with approximately 20 time steps per meter. We measured all runtimes using a single-threaded implementation on an Intel® Core™ i7 2.8GHz with 12GB RAM.

A. Runtime Comparison to the State of the Art

In the first set of experiments, we compare the runtimes of our approach, the approach by van den Berg *et al.* [3] (described in Section IV-B), and a Monte Carlo simulation.

To produce results that are independent of a specific scenario, we considered 120 trajectories with lengths between 25 m and 300 m, each connecting a different set of randomly sampled goal points in a 15 m × 15 m large environment. Furthermore, for each trajectory we individually sampled a map consisting of 20 landmarks. Figure 1 shows four of these trajectories together with the sampled landmark positions.

A detailed comparison of the runtimes of our approach and of the approach by van den Berg *et al.* can be seen in Figure 2. The figure shows the runtimes needed for the complete calculation of the covariances of the expected distributions, which includes the calculation of Σ_t and of the Jacobians defined in (7). Additionally, in darker colors, it shows the

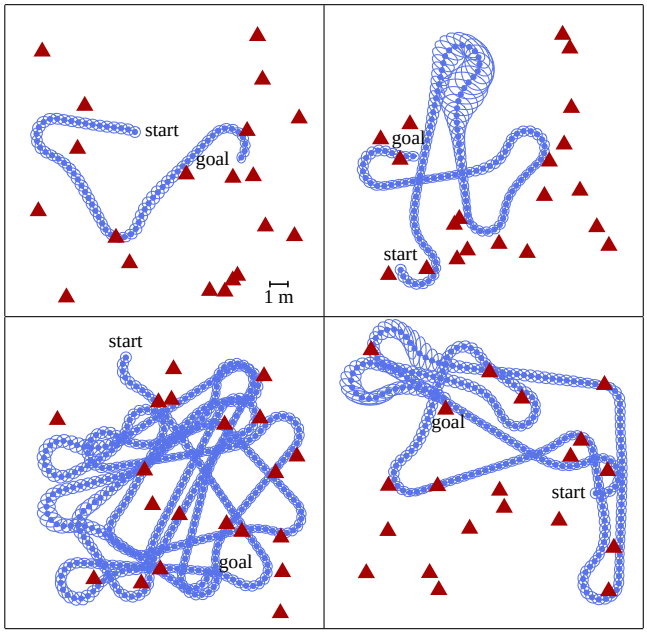

Fig. 1. Randomly sampled trajectories and 99% confidence ellipses of the calculated expected distributions. The depicted trajectories have lengths of 25 m, 50 m, 100 m, and 200 m, respectively (clockwise from upper left). The sampled landmark positions are shown as red triangles.

fractions of the runtimes that were actually spent on the matrix multiplications that our approach speeds up, i.e., the calculation of S_t via M_t as in Algorithm 1 and via J_t as in (21), respectively. As can be seen from the figure, our approach significantly speeds up the matrix multiplication part, and thereby reduces the overall runtime approximately by half. The values of the calculated covariances resulting from the two approaches differed by at most $3.64 \cdot 10^{-12}$, which is within the range of machine precision.

Both our approach and the one by van den Berg *et al.* are orders of magnitude faster than Monte Carlo simulation. For example, estimating the expected distributions for the sampled trajectories with length 100 m took 36.5 sec on average with Monte Carlo simulation. For the same trajectories, our approach spent 0.0117 sec and van den Berg's approach spent 0.0249 sec on average. The runtime of the Monte Carlo simulation depends strongly on the number of simulated episodes used. In our comparison, we used 1,000 episodes, while in practical applications, typically more episodes are needed to achieve the desired accuracy, e.g., Dellaert *et al.* [6] use 5,000 samples for estimating the states of a mobile robot.

B. Application to Landmark Placement

In a second experiment, we applied our approach to the problem of landmark placement. Given the specifications of a mobile robot and a desired trajectory, the goal of landmark placement is to place a minimum number of landmarks around a pre-planned trajectory to guarantee a certain bound on the expected navigation performance during operation. Concretely, we consider the landmark placement approach of Beinhofer *et al.* [2], which aims at bounding the deviation of the robot from the desired trajectory with high confidence.

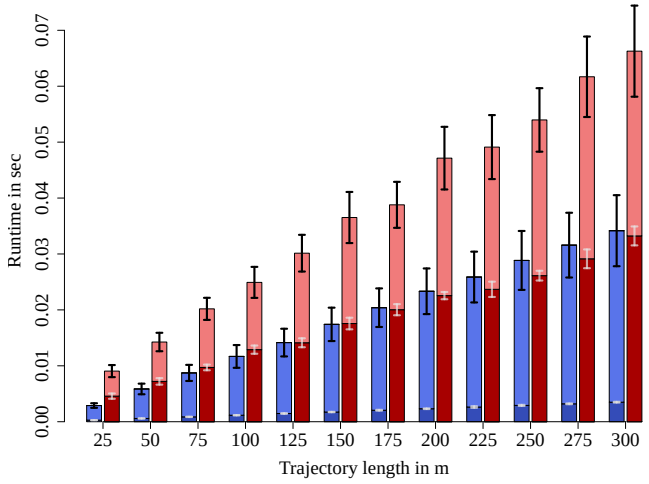

Fig. 2. Comparison between the runtimes (stated in sec) of our approach (blue) and the approach by van den Berg *et al.* (red) on randomly sampled trajectories with lengths between 25 m and 300 m. Shown are the means and 95% error bars. Indicated in dark blue and dark red are the fractions of the runtimes that were spent for the actual matrix multiplications that our approach accelerates.

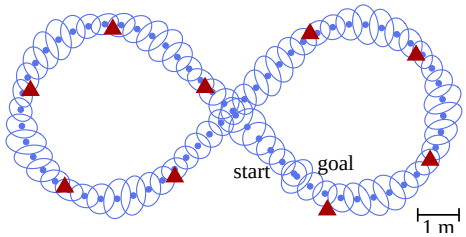

Fig. 3. Landmark positions (red triangles) resulting from a placement procedure that guarantees that the deviation of the robot from the desired trajectory (blue dots) stays below 0.5 m with 99% confidence (blue ellipses).

To calculate the confidence levels, the iterative landmark placement method needs to repeatedly compute expected distributions for different landmark configurations (for details, see [2]). As our experiments showed, most of the runtime in this application is actually spent in the calculation of expected distributions. Therefore, calculating the expected distributions with our approach instead of the approach of van den Berg *et al.* considerably speeds up the whole landmark placement procedure.

Figure 3 shows the placed landmarks for the depicted figure-eight trajectory. The complete landmark placement procedure took 121 sec with our approach and 218 sec with the approach of van den Berg *et al.*

VI. CONCLUSIONS

In this paper, we presented a novel recursive calculation scheme for estimating the expected probability distributions of the states of a mobile robot even before it starts operation. Our approach decouples the calculation of the covariances of the states of the robot and of its localization estimates, which reduces the runtime of the method. In extensive experiments, we showed that our approach significantly outperforms other approaches and that in an application to landmark placement, it considerably speeds up the whole procedure.

APPENDIX

Proof of Lemma 1. We first prove that the means of $\Delta\mathbf{x}_t$ and $\Delta\boldsymbol{\mu}_t$ are zero. Plugging the equation for the LQR control selection (13) into the linearized motion model (5) leads to

$$\Delta\mathbf{x}_t = A_t\Delta\mathbf{x}_{t-1} + B_tL_t\Delta\boldsymbol{\mu}_{t-1} + V_t\,\mathbf{v}_t\,. \tag{24}$$

As $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, Q_t)$, and the expectation is a linear operator, it follows that

$$\mathbb{E}[\Delta\mathbf{x}_t] = A_t\,\mathbb{E}[\Delta\mathbf{x}_{t-1}] + B_tL_t\,\mathbb{E}[\Delta\boldsymbol{\mu}_{t-1}]\,. \tag{25}$$

To derive a similar formula for the expectation of $\Delta\boldsymbol{\mu}_t$, we plug Equation (8) into Equation (11), which yields

$$\begin{aligned}\Delta\boldsymbol{\mu}_t =& A_t\Delta\boldsymbol{\mu}_{t-1} + B_t\Delta\mathbf{u}_t \\ &+ K_t(\Delta\mathbf{z}_t - H_tA_t\Delta\boldsymbol{\mu}_{t-1} + B_t\Delta\mathbf{u}_t)\,.\end{aligned} \tag{26}$$

Plugging in Equations (13) and (6) results in

$$\begin{aligned}\Delta\boldsymbol{\mu}_t =& (A_t + B_tL_t - K_tH_tA_t - K_tB_tL_t)\Delta\boldsymbol{\mu}_{t-1} \\ &+ K_tH_t\Delta\mathbf{x}_t + K_tW_t\,\mathbf{w}_t\,.\end{aligned} \tag{27}$$

Again using the linearity of the expectation operator and using the fact that $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, R_t)$, we get

$$\begin{aligned}\mathbb{E}[\Delta\boldsymbol{\mu}_t] =& (A_t + B_tL_t - K_tH_tA_t - K_tB_tL_t)\,\mathbb{E}[\Delta\boldsymbol{\mu}_{t-1}] \\ &+ K_tH_t\,\mathbb{E}[\Delta\mathbf{x}_t]\,.\end{aligned} \tag{28}$$

Because of the assumption that $\mathbb{E}[\Delta\mathbf{x}_0] = \mathbf{0}$, also $\Delta\boldsymbol{\mu}_0 = \mathbf{0}$, and therefore also $\mathbb{E}[\Delta\boldsymbol{\mu}_0] = \mathbf{0}$. An induction with this as the base case and Equations (25) and (28) as inductive step yields that $\mathbb{E}[\Delta\mathbf{x}_t] = \mathbf{0}$ and $\mathbb{E}[\Delta\boldsymbol{\mu}_t] = \mathbf{0}$.

A second induction with $\Delta\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, S_0)$ and $\Delta\boldsymbol{\mu}_0 = \mathbf{0}$ as base case and with (24) and (27) as inductive step finally shows that the expected distributions are Gaussians. □

Proof of Theorem 1. From Lemma 1, we know that $\mathbb{E}[\Delta\mathbf{x}_t] = \mathbf{0}$. Therefore, the covariance of $p(\Delta\mathbf{x}_t)$ is

$$S_t = \mathrm{Cov}(\Delta\mathbf{x}_t) = \int \Delta\mathbf{x}_t\Delta\mathbf{x}_t^T\; p(\Delta\mathbf{x}_t)\;\mathrm{d}\Delta\mathbf{x}_t\,. \tag{29}$$

Applying the law of total probability [12] on $p(\Delta\mathbf{x}_t)$ yields

$$\begin{aligned}S_t = \int \Delta\mathbf{x}_t\Delta\mathbf{x}_t^T \int & p(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t}) \\ &\cdot p(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}\Delta\mathbf{x}_t\,.\end{aligned} \tag{30}$$

Fubini's theorem [7] allows us to reorder the integrals:

$$\begin{aligned}S_t = \int\int \Delta\mathbf{x}_t\Delta\mathbf{x}_t^T\; & p(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}\Delta\mathbf{x}_t \\ &\cdot p(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\,.\end{aligned} \tag{31}$$

In the following, we use the shorthand notations

$$\mathrm{d}P_{\mathbf{x}} := p(\Delta\mathbf{x}_t \mid \Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}\Delta\mathbf{x}_t\,, \tag{32}$$

$$\mathrm{d}P_{\mathbf{u},\mathbf{z}} := p(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\;\mathrm{d}(\Delta\mathbf{u}_{1:t}, \Delta\mathbf{z}_{1:t})\,. \tag{33}$$

Next, we add a zero to Equation (31):

$$S_t = \int\int \big((\Delta\mathbf{x}_t - \Delta\boldsymbol{\mu}_t)(\Delta\mathbf{x}_t - \Delta\boldsymbol{\mu}_t)^T + \Delta\mathbf{x}_t\Delta\boldsymbol{\mu}_t^T + \Delta\boldsymbol{\mu}_t\Delta\mathbf{x}_t^T - \Delta\boldsymbol{\mu}_t\Delta\boldsymbol{\mu}_t^T\big)\mathrm{d}P_{\mathbf{x}}\,\mathrm{d}P_{\mathbf{u},\mathbf{z}}\,. \tag{34}$$

By definition of the covariance it holds that $\Sigma_t = \int(\Delta\mathbf{x}_t - \Delta\boldsymbol{\mu}_t)(\Delta\mathbf{x}_t - \Delta\boldsymbol{\mu}_t)^T\,\mathrm{d}P_{\mathbf{x}}$, and therefore

$$S_t = \int \Big(\Sigma_t + \int \Delta\mathbf{x}_t\Delta\boldsymbol{\mu}_t^T\,\mathrm{d}P_{\mathbf{x}} + \int \Delta\boldsymbol{\mu}_t\Delta\mathbf{x}_t^T\,\mathrm{d}P_{\mathbf{x}} - \int \Delta\boldsymbol{\mu}_t\Delta\boldsymbol{\mu}_t^T\,\mathrm{d}P_{\mathbf{x}}\Big)\,\mathrm{d}P_{\mathbf{u},\mathbf{z}}\,. \tag{35}$$

As $\Delta\boldsymbol{\mu}_t$ is the expected value of the posterior distribution of $\Delta\mathbf{x}_t$, it is by definition $\Delta\boldsymbol{\mu}_t = \int \Delta\mathbf{x}_t\,\mathrm{d}P_{\mathbf{x}}$, which is independent of $\Delta\mathbf{x}_t$ given the values of $\Delta\mathbf{u}_{1:t}$ and $\Delta\mathbf{z}_{1:t}$. Applying this definition on (35) and using the independence property to reorder the integrals yields

$$S_t = \int \Big(\Sigma_t + \int \Delta\mathbf{x}_t\,\mathrm{d}P_{\mathbf{x}} \int \Delta\mathbf{x}_t^T\,\mathrm{d}P_{\mathbf{x}}\Big)\,\mathrm{d}P_{\mathbf{u},\mathbf{z}}\,. \tag{36}$$

Again using the definition of $\Delta\boldsymbol{\mu}_t$ and the fact that the transpose is a linear transformation, which therefore can be moved out of the integral, yields

$$S_t = \int \Sigma_t + \Delta\boldsymbol{\mu}_t\Delta\boldsymbol{\mu}_t^T\,\mathrm{d}P_{\mathbf{u},\mathbf{z}}\,. \tag{37}$$

In the linearized system that we consider, Σ_t is independent of the values of $\Delta\mathbf{u}_{1:t}$ and $\Delta\mathbf{z}_{1:t}$ (see Equations (9), (10), (12)). Therefore, we get

$$S_t = \Sigma_t + \int \Delta\boldsymbol{\mu}_t\Delta\boldsymbol{\mu}_t^T\,\mathrm{d}P_{\mathbf{u},\mathbf{z}}\,. \tag{38}$$

As the random variable $\Delta\boldsymbol{\mu}_t$ is a deterministic function of the random variables $\Delta\mathbf{u}_{1:t}$ and $\Delta\mathbf{z}_{1:t}$, and as its expectation is $\mathbf{0}$, this results in

$$S_t = \Sigma_t + M_t \tag{39}$$

□

Proof of Corollary 1. Follows from the proof of Theorem 1 by considering the transformations applied to $\Delta\mathbf{x}_t\Delta\boldsymbol{\mu}_t^T$ in Equation (34). □

Proof of Lemma 2. The result follows directly from the construction of the Kalman Filter as described by Kalman [8]. The second equation holds by definition of $\overline{\Sigma}_t$. For linear systems like the one defined in Section III-B, Kalman has proven that the filter mean $\overline{\Delta\boldsymbol{\mu}}_t$ is the minimum mean square error estimator for $\Delta\mathbf{x}_t$ and that the posterior covariance $\overline{\Sigma}_t$ equals the covariance of the estimation error $\Delta\mathbf{x}_t - \overline{\Delta\boldsymbol{\mu}}_t$ of this estimator. □

Proof of Lemma 3. The initial belief in the Kalman Filter for calculating the posterior covariance is deterministically given, therefore Equation (15) holds true. Plugging the LQR control policy from Equation (13) into the recursive update scheme for $\overline{\Delta\boldsymbol{\mu}}_t$ stated in Equation (8) yields

$$\overline{\Delta\boldsymbol{\mu}}_t = (A_t + B_tL_{t-1})\,\Delta\boldsymbol{\mu}_{t-1}\,. \tag{40}$$

As the covariance is a bilinear form [7], this proves Equation (16). To prove Equation (17), we start by plugging Equation (6) into Equation (11), resulting in

$$\Delta\boldsymbol{\mu}_t = \overline{\Delta\boldsymbol{\mu}}_t + K_tH_t(\Delta\mathbf{x}_t - \overline{\Delta\boldsymbol{\mu}}_t) + K_tW_t\mathbf{w}_t, \tag{41}$$

where $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, R_t)$. Again with the bilinearity of the covariance, this yields that

$$M_t = \overline{M}_t + K_tH_t\mathrm{Cov}(\Delta\mathbf{x}_t - \overline{\Delta\boldsymbol{\mu}}_t)H_t^TK_t^T + K_tW_tR_tW_t^TK_t^T\,. \tag{42}$$

Applying Lemma 2 results in

$$M_t = \overline{M}_t + K_tH_t\overline{\Sigma}_tH_t^TK_t^T + K_tW_tR_tW_t^TK_t^T \tag{43}$$

$$= \overline{M}_t + K_t(H_t\overline{\Sigma}_tH_t^T + W_tR_tW_t^T)K_t^T\,. \tag{44}$$

Next, we replace K_t^T according to its definition from Equation (10), leading to

$$M_t = \overline{M}_t + K_t(H_t\overline{\Sigma}_tH_t^T + W_tR_tW_t^T) \cdot (H_t\overline{\Sigma}_tH_t^T + W_tR_tW_t^T)^{-1}H_t\overline{\Sigma}_t \tag{45}$$

$$= \overline{M}_t + K_tH_t\overline{\Sigma}_t\,, \tag{46}$$

which finishes the proof. □

REFERENCES

[1] Y. Bar-Shalom, T. Kirubarajan, and X. Li. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2002.

[2] M. Beinhofer, J. Müller, and W. Burgard. Effective landmark placement for accurate and reliable mobile robot navigation. *Robotics and Autonomous Systems (RAS)*, 61(10):1060 – 1069, 2013.

[3] J. van den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. In *Proc. of Robotics: Science and Systems (RSS)*, 2010.

[4] J. van den Berg, S. Patil, R. Alterovitz, P. Abbeel, and K. Goldberg. LQG-based planning, sensing, and control of steerable needles. In *Proc. of the Int. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2010.

[5] D. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 3rd edition, 2005.

[6] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1999.

[7] O. Kallenberg. *Foundations of modern probability*. Springer Verlag, 2002.

[8] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *ASME Journal of Basic Engineering*, (82):35–45, 1960.

[9] F. Mastrogiovanni, A. Sgorbissa, and R. Zaccaria. How the location of the range sensor affects EKF-based localization. *Journal of Intelligent & Robotic Systems*, 68(2):121–145, 2012.

[10] S. Patil, J. van den Berg, and R. Alterovitz. Motion planning under uncertainty in highly deformable environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2011.

[11] N. Roy, W. Burgard, D. Fox, and S. Thrun. Coastal navigation: Mobile robot navigation with uncertainty in dynamic environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1999.

[12] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2006.

[13] M. Vitus and C. Tomlin. Sensor placement for improved robotic navigation. In *Proc. of Robotics: Science and Systems (RSS)*, 2010.

[14] M. Vitus and C. Tomlin. Closed-loop belief space planning for linear, gaussian systems. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.