Effective Landmark Placement for Accurate and Reliable Mobile Robot Navigation

Maximilian Beinhofer, Jörg Müller, Wolfram Burgard

Department of Computer Science, University of Freiburg, D-79110 Freiburg, Germany

Abstract

Being able to navigate accurately is one of the fundamental capabilities of a mobile robot to effectively execute a variety of tasks including docking, transportation, and manipulation. As real-world environments often contain changing or ambiguous areas, existing features can be insufficient for mobile robots to establish a robust navigation behavior. A popular approach to overcome this problem and to achieve accurate localization is to use artificial landmarks. In this paper, we consider the problem of optimally placing such artificial landmarks for mobile robots that repeatedly have to carry out certain navigation tasks. Our method aims at finding the minimum number of landmarks for which a bound on the maximum deviation of the robot from its desired trajectory can be guaranteed with high confidence. The proposed approach incrementally places landmarks utilizing linearized versions of the system dynamics of the robot, thus allowing for an efficient computation of the deviation guarantee. We evaluate our approach in extensive experiments carried out both in simulation and with real robots. The experiments demonstrate that our method outperforms other approaches and is suitable for long-term operation of mobile robots.

Keywords: Localization, Autonomous Navigation, Service Robots

1. Introduction

One of the major challenges for mobile service robots is safe and reliable autonomous navigation. Robots navigating autonomously generally deviate from their desired trajectory due to uncertainty in both motion and position. Usually, they are equipped with on-board sensors to estimate the deviation and react according to a feedback control law. According to our experience, for example in logistics and transportation applications, a substantial degree of accuracy is required for localization, as the navigation tasks of the robot usually include maneuvers like pick-and-place or docking. Today's versatile production sites and warehouses often contain areas that frequently undergo changes and also may contain ambiguous areas. In such environments, the restriction to existing features for orientation can be insufficient for establishing safe and reliable navigation. Many practical applications therefore rely on artificial landmarks placed along the trajectories taken by the robot to allow for accurate localization [8, 10]. Placing artificial landmarks is often expensive and at the same time the computational power of the robot is limited which imposes substantial limits on the number of landmarks that can be placed. Accordingly, it is desired to select the smallest possible number of landmark positions which still guarantees the required accuracy in navigation.

In this paper, we present a novel algorithm for landmark placement, which computes a landmark configuration such that the deviation of the robot from its desired trajectory stays below a user-defined threshold $d_{\max}$ with high confidence. To check if the guarantee holds, we use the specific properties of the robot and its navigation task in the landmark placement algorithm. Our method works in two stages. In the first stage, it uses linearized motion and observation models of the robot to efficiently place landmarks in an incremental fashion. This stage aims at placing the smallest number of landmarks for which the deviation guarantee holds. In the second stage, the algorithm employs a Monte Carlo simulation for the computed landmark configuration to validate that the guarantee also holds for the possibly non-linear models.

Our approach has several characteristics which make it especially useful for mobile robot navigation. It can deal with arbitrary trajectories, and the maximum allowed deviation of the robot can be defined individually for every part of the trajectories. Taking into account the properties of the individual robotic system results in customized landmark sets: while high-precision robots need only a few landmarks for reaching the deviation guarantee, low-cost systems typically require more landmarks. As our placement algorithm efficiently evaluates the guarantee using linear models, it can deal even with large instances of the landmark placement problem (i.e., long trajectories). Note that our incremental method simultaneously determines the number and positions of the landmarks needed to meet the desired guarantee.

This paper is organized as follows. After discussing related work in the following section, we formalize the problem definition in Section 3. In Section 4, we describe the prediction of the deviation from the trajectory in linearized systems. Afterwards, in Section 5, we present our incremental landmark placement algorithm. In Section 6 we give a theoretical evaluation of our method. Finally, we provide extensive experiments in which we evaluate the algorithm in simulations and in real-world applications.

2. Related Work

In the past, the problem of finding an optimal set of landmark positions has been addressed from several points of view. Salas and Gordillo [22] consider it in terms of the art gallery problem. They use simulated annealing to find a landmark set which maximizes the area in which a robot has a clear line of sight to at least one landmark. Erickson and LaValle [9] consider the same problem for colored landmarks. They add the constraint that from no position in the map two landmarks of the same color may be visible. They give bounds for the minimum number of colors needed. Sala *et al.* [21] extend this problem to select landmark positions so that at every position in the map, at least k landmarks are observable. Rupp and Levi [20] select landmark positions on the walls of an indoor environment close to a given set of localization points. They use geometrical insights to find the landmark locations. Unlike these methods, which assume a deterministic robot behavior, our approach explicitly models the noise of the sensors and actuators of the robot.

Jourdan and Roy [11] consider a fixed set of possible target positions. They place sensors on the walls of buildings to minimize the average position error bound in the sensor network. Likewise, Meyer-Delius *et al.* [16] present an approach that is independent of the trajectory taken by the robot. They increase the localization accuracy of a system already equipped with a landmark-independent sensor (e.g., a laser range finder) by placing additional landmarks in the environment. In contrast to these methods, our approach takes into account the full specification of the robot and its navigation task.

Like our approach, Vitus and Tomlin [25] consider the full problem specification to place sensors in the environment. They approximate the a priori covariances with the a posteriori covariances of the most likely run of the robot. Similar to our approach, van den Berg *et al.* [4] evaluate sensor positions using the exact a priori distributions in a linearized system. As they focus mainly on path planning, they restrict themselves to randomly sampled positions of a single sensor. Our previous work [2] selects landmarks maximizing the mutual information between the sensor readings and the states of the robot. It solely applies Monte Carlo simulations to estimate the a priori distributions, which makes it computationally more demanding.

While all of the approaches above place artificial landmarks or sensors before the operation of the robot, the following approaches decide whether to utilize observed landmarks during operation. In contrast to our method, their decisions are based on a posteriori distributions, i.e., the information already gathered by the robot. Thrun [24] selects the subset of the observed landmarks for localization which minimizes the average posterior localization error. Lerner *et al.* [15] use semidefinite programming to select landmarks that minimize the trace of the pose covariance of a moving camera. Strasdat *et al.* [23] and Zhang *et al.* [26] both consider landmark selection in the context of the simultaneous localization and mapping (SLAM) problem. Strasdat *et al.* use reinforcement learning to create a landmark selection policy whereas Zhang *et al.* minimize the entropy of the a posteriori distributions.

In contrast to the above-mentioned approaches, our method optimizes the positions of the landmarks so that the robot stays within a user-defined region around the trajectory with high a priori probability.

3. Problem Definition

We consider the problem of placing landmarks for localization and control of a mobile robot. We assume the time to be discretized into steps of equal duration. At each time step t the state of the robot is defined by a vector $\mathbf{x}_t \in \mathcal{X}$, which changes over time according to the stochastic motion model

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

where $\mathbf{u}_t \in \mathcal{U}$ is the control command at time t. Thereby, the motion is disturbed by Gaussian noise

$\mathbf{m}_t \sim \mathcal{N}(\mathbf{0}, M_t)$. For self-localization, we assume that the robot is equipped with a sensor taking measurements of a set of landmarks $\mathcal{A} = \{\boldsymbol{\ell}_1, ..., \boldsymbol{\ell}_n\}$ according to the measurement function

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{n}_t, \mathcal{A}) \tag{2}$$

where the sensor signal is disturbed by Gaussian noise $\mathbf{n}_t \sim \mathcal{N}(\mathbf{0}, N_t)$. The covariances M_t and N_t model the uncertainty in the motion and the measurements, respectively.

We define a navigation task as a trajectory that the robot should follow. A trajectory $\mathcal{T} = (\mathbf{x}_0^\star, \mathbf{u}_0^\star), \ldots, (\mathbf{x}_T^\star, \mathbf{u}_T^\star)$ can be considered as a series of states and desired controls the robot should execute to reach these states. In this navigation task, we assume that the trajectory will be executed using a linear-quadratic regulator (LQR) [5] feedback controller. At each time step t the LQR controller selects the control command $\mathbf{u}_t$ which minimizes the quadratic cost function

$$\mathbb{E}\Big[\sum_{k=t}^{T}((\mathbf{x}_k - \mathbf{x}_k^\star)^T C(\mathbf{x}_k - \mathbf{x}_k^\star) + (\mathbf{u}_k - \mathbf{u}_k^\star)^T D(\mathbf{u}_k - \mathbf{u}_k^\star))\Big], \tag{3}$$

where C and D are positive-definite weight matrices.

The localization uncertainty and, as a result, also the deviation from the desired trajectory strongly depend on the specific configuration of landmarks $\mathcal{A} = \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_n\}$ which are observed during operation. Our approach selects landmarks $\boldsymbol{\ell}_i \in \mathcal{L}$ from a continuous space of possible landmark locations. We evaluate the quality of a landmark configuration based on the deviation of the (real) state $\mathbf{x}_t$ from the desired state $\mathbf{x}_t^\star$ at each time step t (ignoring the control part $\mathbf{u}_{0:T}^\star$ of the trajectory). In particular, we consider the Euclidean distance between the part of the state $\mathbf{x}_t^{\text{pos}}$ describing the position of the robot and $\mathbf{x}_t^{\star\text{pos}}$. We focus on limiting the deviation

$$d^{\text{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star) = \|\mathbf{x}_t^{\text{pos}} - \mathbf{x}_t^{\star\text{pos}}\|_2 \tag{4}$$

of the robot from its trajectory at all time steps $t \in [0, T]$. Note that limiting d^{pos} implicitly limits the other relevant parts of the state, too. A large error in rotation, for example, would result in increasing deviations of the positions in consecutive time steps, and is therefore restricted. Our approach aims at finding the landmark configuration $\mathcal{A}$ with the fewest elements for which the *deviation guarantee*

$$\forall t \in [0, T] : \; p\left(d^{\text{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star) \le d_{\max}(\mathbf{x}_t^\star) \mid \mathcal{A}\right) \ge p_{\min} \tag{5}$$

holds. This guarantee ensures that the probability of deviating at most $d_{\max}$ from the desired trajectory is at least $p_{\min}$. Note that $d_{\max}$ can be either a globally constant value or depend on the position or time.

4. Predicting the Deviation from the Trajectory

To validate the guarantee (5) for a certain landmark configuration $\mathcal{A}$, we need to compute $p\left(d^{\text{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star) \le d_{\max}(\mathbf{x}_t^\star) \mid \mathcal{A}\right)$. For this, we consider the a priori probability distribution

$$\begin{aligned} p(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathcal{A}) = \int\int p(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t}, \mathcal{A}) \\ \cdot\, p(\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t} \mid \mathcal{A})\, \mathrm{d}\mathbf{u}_{0:t-1}\, \mathrm{d}\mathbf{z}_{1:t}\,, \end{aligned} \tag{6}$$

which averages over the observations $\mathbf{z}_{1:t}$ and controls $\mathbf{u}_{0:t-1}$ that are not yet available during landmark placement.

For general non-linear systems, the a posteriori distributions $p(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathbf{u}_{1:t-1}, \mathbf{z}_{1:t}, \mathcal{A})$ can be used to estimate the a priori distributions (6) via Monte Carlo simulation by sampling observations and controls and averaging over numerous runs [2]. However, this is computationally expensive for large instances of the landmark placement problem.

4.1. A-Priori State Estimation in Linearized, Gaussian Systems

In the main part of our landmark placement algorithm, we locally linearize the system around the desired trajectory and approximate all distributions by Gaussians. This allows for an analytical evaluation of the guarantee (5), making it substantially more efficient than Monte Carlo simulations. Linearizing the motion model (1) and the sensor model (2) around the desired trajectory ($\mathbf{x}_{0:T}^\star$, $\mathbf{u}_{0:T}^\star$) by first-order Taylor expansion leads to

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}^\star, \mathbf{u}_{t-1}^\star, \mathbf{0}) + A_t(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^\star) \\ &\quad + B_t(\mathbf{u}_{t-1} - \mathbf{u}_{t-1}^\star) + V_t\,\mathbf{m}_t\,, \end{aligned} \tag{7}$$

$$\mathbf{z}_t = h(\mathbf{x}_t^\star, \mathbf{0}, \mathcal{A}) + H_t(\mathbf{x}_t - \mathbf{x}_t^\star) + W_t\,\mathbf{n}_t\,, \tag{8}$$

with the Jacobians

$$\begin{aligned} A_t &= \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}_{t-1}^\star, \mathbf{u}_{t-1}^\star, \mathbf{0}),\; B_t = \frac{\partial f}{\partial \mathbf{u}}(\mathbf{x}_{t-1}^\star, \mathbf{u}_{t-1}^\star, \mathbf{0}), \\ V_t &= \frac{\partial f}{\partial \mathbf{m}}(\mathbf{x}_{t-1}^\star, \mathbf{u}_{t-1}^\star, \mathbf{0}), \\ H_t &= \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}_t^\star, \mathbf{0}, \mathcal{A}),\; W_t = \frac{\partial h}{\partial \mathbf{n}}(\mathbf{x}_t^\star, \mathbf{0}, \mathcal{A}). \end{aligned} \tag{9}$$

In this linearized system, the Gaussian a posteriori distribution $p(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathbf{u}_{1:t-1}, \mathbf{z}_{1:t}, \mathcal{A}) \sim \mathcal{N}(\boldsymbol{\mu}_t - \mathbf{x}_t^\star, P_t)$

of the deviation from the trajectory can be computed recursively using a Kalman filter [1]. The Kalman filter propagates a given initial Gaussian distribution $p(\mathbf{x}_0 - \mathbf{x}_0^\star \mid \mathcal{A}) \sim \mathcal{N}(\boldsymbol{\mu}_0 - \mathbf{x}_0^\star, P_0)$ according to the actual control commands in the *motion update*

$$\begin{aligned}\bar{\boldsymbol{\mu}}_t - \mathbf{x}_t^\star &= A_t(\boldsymbol{\mu}_{t-1} - \mathbf{x}_{t-1}^\star) + B_t(\mathbf{u}_{t-1} - \mathbf{u}_{t-1}^\star)\\ \bar{P}_t &= A_t P_{t-1} A_t^T + V_t M_t V_t^T \,. \qquad (10)\end{aligned}$$

and according to the measurements in the *observation update*

$$\begin{aligned}K_t &= \bar{P}_t H_t^T (H_t \bar{P}_t H_t^T + W_t N_t W_t^T)^{-1} \qquad (11)\\ \boldsymbol{\mu}_t - \mathbf{x}_t^\star &= \bar{\boldsymbol{\mu}}_t - \mathbf{x}_t^\star + K_t(\mathbf{z}_t - h(\mathbf{x}_t^\star, \mathbf{0}, \mathcal{A}) - H_t(\bar{\boldsymbol{\mu}}_t - \mathbf{x}_t^\star))\\ P_t &= (I - K_t H_t)\bar{P}_t \,. \qquad (12)\end{aligned}$$

Note that the covariance P_t and the Kalman gain K_t depend, via the Jacobians, on $\mathbf{x}_{0:t}^\star$ and $\mathbf{u}_{0:t-1}^\star$ but not on the actual values of $\mathbf{u}_{0:t-1}$ and $\mathbf{z}_{1:t}$ (see (10), (11), (12)). Therefore they can be calculated before the robot starts operation (a priori).

The minimization of the expected deviation from the desired trajectory (3) in the LQR controller can also be solved a priori, linearly relating the control command $\mathbf{u}_t$ to the estimated state $\boldsymbol{\mu}_t$ via a feedback matrix L_t:

$$\mathbf{u}_t - \mathbf{u}_t^\star = L_t(\boldsymbol{\mu}_t - \mathbf{x}_t^\star)\,. \qquad (13)$$

L_t depends on the a priori known Jacobians (9) and the weight matrices (3) and is derived explicitly in [5].

As described above, we express the whole navigation algorithm, which consists of executing a motion command, making an observation, localizing, and selecting the next motion command depending on the localization, by linear functions. For this linear navigation system, van den Berg *et al.* [3] have proven that the a priori joint distribution of $\mathbf{x}_t$ and $\boldsymbol{\mu}_t$ is a Gaussian

$$\begin{bmatrix}\mathbf{x}_t - \mathbf{x}_t^\star\\ \boldsymbol{\mu}_t - \mathbf{x}_t^\star\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, R_t = \begin{bmatrix}S_t & \mathrm{Cov}(\mathbf{x}_t, \boldsymbol{\mu}_t)\\ \mathrm{Cov}(\mathbf{x}_t, \boldsymbol{\mu}_t)^T & U_t\end{bmatrix}\right),$$

and that its covariance R_t can be computed recursively by

$$R_0 = \begin{bmatrix}P_0 & 0\\ 0 & 0\end{bmatrix},\ R_t = F_t R_{t-1} F_t^T + G_t \begin{bmatrix}M_t & 0\\ 0 & N_t\end{bmatrix} G_t^T,$$

with

$$\begin{aligned}F_t &= \begin{bmatrix}A_t & B_t L_{t-1}\\ K_t H_t A_t & A_t + B_t L_{t-1} - K_t H_t A_t\end{bmatrix}, \qquad (14)\\ G_t &= \begin{bmatrix}V_t & 0\\ K_t H_t V_t & K_t W_t\end{bmatrix}. \qquad (15)\end{aligned}$$

R_t only depends on a priori known variables, namely the Jacobians (9), the Kalman gain (11), and the feedback matrix (13). These variables can be computed a priori since we linearize the models around the (a priori known) desired states $\mathbf{x}_{0:T}^\star$ and not around the (a priori unknown) estimates $\boldsymbol{\mu}_{0:T}$, as it is done for example in the extended Kalman filter [1].

4.2. Evaluation of the Deviation Guarantee

In the linearized system we can efficiently check whether the deviation guarantee (5) holds. Let S_t^{pos} be the part of the a priori covariance S_t of $p(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathcal{A})$ corresponding to the position of the robot. The length $a_t(\mathcal{A})$ of the major semi-axis of the $p_{\min}$-confidence ellipsoid of S_t^{pos} can be calculated using

$$a_t(\mathcal{A}) = c_{\min}\sqrt{\lambda_t}\,, \qquad (16)$$

where λ_t is the largest eigenvalue of S_t^{pos} and $c_{\min}$ is a scaling factor corresponding to $p_{\min}$ via the regularized Gamma function as described in [3]. If $a_t(\mathcal{A}) \le d_{\max}$, then the $p_{\min}$-ellipsoid of S_t^{pos} is inside a circle with radius $d_{\max}$ and guarantee (5) holds for the linearized system. Note that this test is a conservative approximation and is exact if the $p_{\min}$-ellipsoid is a sphere.

4.3. Visibility of Landmarks

When considering robots with a limited sensor range or occlusions due to objects inside the field of view, the non-linearity of the sensor model at these borders induces a large discrepancy between the real model and its linearization. To avoid this, we estimate for every landmark if the robot will observe it at time t, following the approach of Vitus and Tomlin [25]. We consider a landmark as visible at time t only if it is $p_{\min}$-*visible*, i.e., it is visible from every pose inside the $p_{\min}$-ellipsoid of S_t around $\mathbf{x}_t^\star$. If a landmark configuration satisfies the guarantee (5) when only $p_{\min}$-visible landmarks are observed, it also satisfies it when using all visible landmarks.

If the environment of the robot is a planar free space and the robot has a circular field of view, we can check analytically if a landmark is $p_{\min}$-visible. For other types of scenarios, we apply an approximative check utilizing a sigma-point method.

4.3.1. Visibility in Free Space

For the two-dimensional case (i.e., $\mathbf{x}_t^{\mathrm{pos}} = [x, y]^T$) without occlusions or other restrictions, and for a robot with a circular field of view with radius r, there exists a closed-form solution to checking if a given landmark $\boldsymbol{\ell}$ is $p_{\min}$-visible. Because of the circular field of view, the orientation of the robot does not matter for checking the visibility of landmarks. Therefore, for the deviation guarantee to hold it suffices to check the $p_{\min}$-visibility using the $p_{\min}$-ellipse of S_t^{pos} instead of the $p_{\min}$-ellipsoid of S_t. Consider the $p_{\min}$-ellipse of S_t^{pos} centered at the origin of the coordinate system. We apply a principal axis transformation on the ellipse so that afterwards its semi-axes lie on the axes of the coordinate system. Applying this transformation on S_t^{pos} yields a diagonal matrix $S_t^{\text{pos}\prime} = \text{diag}(\lambda_1, \lambda_2)$ with the diagonal elements identical to the eigenvalues of the matrix. Any point $\mathbf{x}' = [x_1', x_2']^T$ on the transformed ellipse $\mathcal{E}$ can then be described as $\mathbf{x}' = c_{\min}\,\text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2})\,\mathbf{x}$ for a point $\mathbf{x} = [x_1, x_2]^T$ on the unit circle C and the scaling factor $c_{\min}$ from (16). To check if landmark $\boldsymbol{\ell}$ is $p_{\min}$-visible given an a priori estimate $(\mathbf{x}_t^\star, S_t)$, we apply the same principal axis transformation also on the relative position $(\boldsymbol{\ell} - \mathbf{x}_t^\star)$ of the landmark, resulting in $\boldsymbol{\ell}' = [\ell_1', \ell_2'] = c_{\min}\,\text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2})\,(\boldsymbol{\ell} - \mathbf{x}_t^\star)$. Checking if $\boldsymbol{\ell}$ is $p_{\min}$-visible means checking if

$$
\begin{aligned}
&\max_{\mathbf{x}'\in\mathcal{E}} \|\mathbf{x}' - \boldsymbol{\ell}'\|_2 \le r \\
\Leftrightarrow\ &\max_{\mathbf{x}\in C} \|c_{\min}\,\text{diag}\left(\sqrt{\lambda_1}, \sqrt{\lambda_2}\right)\mathbf{x} - \boldsymbol{\ell}'\|_2 \le r \\
\Leftrightarrow\ &\max_{x_1\in[-1,1],\,\text{sgn}\in\{-1,1\}} \left(\left(c_{\min}\sqrt{\lambda_1}x_1 - \ell_1'\right)^2 \right. \\
&\qquad \left. + \left(c_{\min}\sqrt{\lambda_2}\,\text{sgn}\sqrt{1 - x_1^2} - \ell_2'\right)^2 - r^2\right) \le 0\,.
\end{aligned}
$$

Applying a distinction of cases for sgn, we set the derivative with respect to x_1 of the function inside the max-operator to 0 and reorder the resulting equation. This yields a quartic term, which can be solved analytically in an efficient way.

4.3.2. Visibility in Structured Environments

For general sensor models and environments in which structures like walls restrict the field of view of the sensor (like in the framework proposed in [19]), the analytical solution to finding the $p_{\min}$-visible landmarks is not applicable. In these cases, we approximate the solution by selecting a set $\mathcal{S}$ of $(2\dim(\mathbf{x}_t^\star) + 1)$ poses on the $p_{\min}$-ellipsoid of S_t similar to the set of sigma-points used in the unscented Kalman filter [12]. $\mathcal{S}$ consists of the center of the $p_{\min}$-ellipsoid and the $2\dim(\mathbf{x}_t^\star)$ poses in which the semi-axes leave the ellipsoid. Knowing the setup of the environment, we can check for a given landmark $\boldsymbol{\ell}$ and a given pose $s \in \mathcal{S}$ if the sensor of the robot would be able to observe $\boldsymbol{\ell}$ when positioned at s. If this check evaluates to true for all $s \in \mathcal{S}$, we consider $\boldsymbol{\ell}$ as $p_{\min}$-visible.

5. Incremental Landmark Placement Algorithm

Our landmark placement approach aims at minimizing the number of landmarks that have to be placed for the deviation guarantee to hold. Since the dimensionality of the search space grows with the length of the trajectory, in general, globally searching for the optimal landmark configuration is computationally intractable. However, using an incremental placement algorithm, we can efficiently find an approximate solution to the landmark placement problem.

5.1. Landmark Placement for the Linearized System

In a first stage, our algorithm employs the linearized system to incrementally place landmarks. Considering linearized Gaussian models is beneficial because the a priori distributions can be efficiently calculated analytically as described in Section 4.1. The objective of our approach is to minimize the number of landmarks needed for the deviation guarantee to hold on the whole trajectory $(\mathbf{x}_{0:T}^\star, \mathbf{u}_{0:T}^\star)$. We approximate this minimum by maximizing the number of time steps for which every additional landmark guarantees (5). Let

$$
t_{\max}(\mathcal{A}) = \max\{t \mid a_s(\mathcal{A}) \le d_{\max}\ \forall s \le t\} \tag{17}
$$

be the maximum time step for which the landmark set $\mathcal{A}$ guarantees (5) in the linearized system for the first part of the trajectory $(\mathbf{x}_{0:t_{\max}}^\star, \mathbf{u}_{0:t_{\max}}^\star)$. $t_{\max}(\mathcal{A})$ obviously depends on $\mathbf{x}_{0:T}^\star$ and $\mathbf{u}_{0:T}^\star$, but for readability, we drop this dependency in the formula. In every iteration our algorithm adds the landmark $\boldsymbol{\ell}_{\text{new}}$ which maximizes $t_{\max}$ to the already selected set of landmarks $\mathcal{A}$. In some cases, one additional landmark is not enough to increase $t_{\max}$. This can happen for example if $d_{\max}(\mathbf{x}_{t_{\max}+1}^\star)$ is chosen considerably smaller than $d_{\max}(\mathbf{x}_{t_{\max}}^\star)$. In these cases, the algorithm selects the landmark which minimizes $a_{t_{\max}}(\mathcal{A})$ instead. Reducing $a_{t_{\max}}(\mathcal{A})$ increases the likelihood that in the next step a landmark can be found which increases $t_{\max}$ again (see (17)). Algorithm 1 describes the incremental landmark placement for the linearized system. Note that in the algorithm, the argmax and the argmin operator can be implemented in several ways depending on the structure of the space $\mathcal{L}$. See Section 7.1 for details on our implementation.

5.2. Monte Carlo Validation

In a second stage, we check the computed landmark configuration $\mathcal{A}$ for the deviation guarantee via Monte Carlo simulation using the real (possibly non-linear, non-Gaussian) models [7]. This is necessary to account for approximation errors due to the linearization and

Algorithm 1 Landmark Placement for the Linearized System

```
Input: Navigation task, space of landmark locations $\mathcal{L}$
Output: Landmark configuration $\mathcal{A}$
  $\mathcal{A} \leftarrow \varnothing$
  $\tau \leftarrow 0$
  while $\tau < T$ do
    $\boldsymbol{\ell}_{\text{new}} \leftarrow \operatorname*{argmax}_{\boldsymbol{\ell} \in \mathcal{L}} t_{\max}(\mathcal{A} \cup \{\boldsymbol{\ell}\})$
    $\tau_{\text{new}} \leftarrow t_{\max}(\mathcal{A} \cup \{\boldsymbol{\ell}_{\text{new}}\})$
    if $\tau_{\text{new}} = \tau$ then
      $\boldsymbol{\ell}_{\text{new}} \leftarrow \operatorname*{argmin}_{\boldsymbol{\ell} \in \mathcal{L}} a_{t_{\max}}(\mathcal{A} \cup \{\boldsymbol{\ell}\})$
    end if
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{\boldsymbol{\ell}_{\text{new}}\}$
    $\tau \leftarrow \tau_{\text{new}}$
  end while
  return $\mathcal{A}$
```

the Gaussian assumption. The Monte Carlo simulation samples robot states $\mathbf{x}_{0:T}$, controls $\mathbf{u}_{0:T}$, and observations $\mathbf{z}_{1:T}$ of the landmarks in $\mathcal{A}$ and counts the number of time steps t in which $d^{\text{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star) \le d_{\max}(\mathbf{x}_t^\star)$, as required in guarantee (5). Averaging over numerous runs yields an estimate p_{MC} of $p_{\min}$ for which the deviation guarantee in the real system holds. If $p_{\text{MC}} < p_{\min}$, one can use arbitrary heuristics to place additional landmarks. For example, one could run our algorithm for increased values of $p_{\min}$ or decreased values of $d_{\max}$.

5.3. Continuous Operation on Round Trips

For round-trip tasks, for which $\mathbf{x}_0^\star = \mathbf{x}_T^\star$, Algorithm 1 can be used for finding a landmark set that guarantees the error bound for multiple successive executions of the task. This can be achieved by designing the part of the initial a priori covariance S_0^{pos} corresponding to the position of the robot so that the $p_{\min}$-ellipsoid of S_T^{pos} at the final time step is inside the $p_{\min}$-ellipsoid of S_0^{pos}. If this property holds, then, when reaching the goal, the robot is inside the $p_{\min}$-ellipsoid of S_0^{pos} with probability greater than or equal to $p_{\min}$. Therefore when starting a next run of the same navigation task, the deviation guarantee is still satisfied.

6. Relation between the Deviation Guarantee and the Localization Uncertainty

In mobile robotics, often the trace of the a posteriori covariance $\operatorname{tr}(P_t)$ is used as a measure for the localization uncertainty [25, 15]. In this section, we show that by enforcing the deviation guarantee, our approach also guarantees a bound on $\operatorname{tr}(P_t)$.

As shown by Kalman [13], the a posteriori covariance P_t calculated by the Kalman filter is also the covariance of its estimation error, i.e., $P_t = \operatorname{Cov}(\mathbf{x}_t - \boldsymbol{\mu}_t \mid \mathcal{A})$. Using this together with the basic property of Kalman filtering that $\boldsymbol{\mu}_t$ is the *minimum mean square error estimator* for $\mathbf{x}_t$, we get the following relation between the traces of the matrices:

Lemma. $\operatorname{tr}(S_t) \ge \operatorname{tr}(P_t)$.

Proof. The Kalman filter is constructed so that in the linearized system, its mean $\boldsymbol{\mu}_t$ is the *minimum mean square error estimator* for $\mathbf{x}_t$ (see Kalman [13]). Therefore it holds that for all estimators $\hat{\mathbf{x}}_t$ of $\mathbf{x}_t$

$$\operatorname{tr}(\operatorname{Cov}(\mathbf{x}_t - \hat{\mathbf{x}}_t \mid \mathcal{A})) \ge \operatorname{tr}(\operatorname{Cov}(\mathbf{x}_t - \boldsymbol{\mu}_t \mid \mathcal{A})) = \operatorname{tr}(P_t)\,.$$

As $S_t = \operatorname{Cov}(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathcal{A})$ and $\mathbf{x}_t^\star$ is a valid estimator, it follows that

$$\operatorname{tr}(S_t) = \operatorname{tr}(\operatorname{Cov}(\mathbf{x}_t - \mathbf{x}_t^\star \mid \mathcal{A})) \ge \operatorname{tr}(P_t)\,.$$

□

This means that if the state $\mathbf{x}_t$ consists only of the position of the robot, i.e., $S_t = S_t^{\text{pos}}$, then our algorithm, which restricts the a priori distribution of $\mathbf{x}_t$, also restricts the localization uncertainty $\operatorname{tr}(P_t)$:

$$\dim(\mathbf{x}_t)\, d_{\max}^2 \ge c_{\min}^2 \operatorname{tr}(S_t^{\text{pos}}) = c_{\min}^2 \operatorname{tr}(S_t) \ge c_{\min}^2 \operatorname{tr}(P_t)\,,$$

where the first inequality results from (16), and $c_{\min}$ is the scaling factor defined in Section 4.2.

7. Experimental Results

We evaluated our landmark placement algorithm and compared it to other landmark placement approaches in extensive experiments both in simulation and with real robots.

7.1. Experimental Setup

In our experiments we considered wheeled robots navigating on a plane. As the most common drive types used in industry are (non-holonomic) differential drive robots and holonomic robots equipped with Mecanum wheels, we carried out our experiments with robots of these types. For self-localization we considered three different types of sensors detecting uniquely identifiable landmarks: a *range-only* sensor, measuring only the distance to the landmarks, a *bearing-only* sensor, measuring only the relative angle between the robot and the landmarks, and a *range-and-bearing* sensor, measuring both. Hence, in the following experiments all motion models and all sensor models have non-linear components.

We evaluated two different kinds of landmark placement depending on the environment the robot operates in. In the free-space setting without any obstacles in the environment of the robot, we considered the complete two-dimensional plane the robot was navigating on as space $\mathcal{L}_{\text{free}}$ of possible landmark locations. We also evaluated our approach in structured environments which were defined by walls. Here, we allowed landmark placement only on the surfaces of predefined walls resulting in a one-dimensional space $\mathcal{L}_{\text{walls}}$ of possible landmark locations.

We implemented the argmax and argmin operators used in Algorithm 1 in a two-stage procedure which is robust to local optima. In the first step, we discretized the relevant part of $\mathcal{L}$ that is observable from the trajectory and evaluated each of these points. Around the optimum on the discrete point set, we then did a fine search with Powell's method [18] in the second step.

7.2. *Placement in Free Space*

In the first set of experiments, we considered environments without obstacles and allowed landmark places in the complete two-dimensional plane $\mathcal{L}_{\text{free}}$. For these experiments, we utilized the analytical method for checking landmark visibility in the placement algorithm as described in Section 4.3.1. For all types of sensors, we assumed a circular field of view around the robot with radius 2 m. We evaluated our algorithm on five navigation tasks T1-T5 (see Fig. 1 and 2) for a differential drive robot. We simulated every task for all three sensor models, resulting in 15 experiments. Fig. 1 shows the landmarks our algorithm computed for the three sensor models in the first task T1, together with a priori and a posteriori distributions.

Fig. 2 depicts the landmark configurations and a priori distributions for the other four tasks T2-T5 for a range-only sensor. For all trajectories, we set $p_{\min} = 99\%$ and $d_{\max} = 0.5$ m. For the pick-and-place task T5, we changed $d_{\max}$ to 0.2 m in the pick-up and in deposit zones (gray rectangles). Because of the high accuracy necessary to fulfill this task, we simulated a more precise robotic system than for the other tasks, i.e., we scaled down the noise values of the motion model and the sensor model of the robot.

7.2.1. *Influence of the Sensor Model*

As can be seen in Fig. 1, the number of landmarks our algorithm computes and their configuration strongly depend on the chosen sensor model. For the range-only sensor, the landmarks tend to be further away from the trajectory than for the other two sensor models. The numbers of landmarks needed are stated in the first row of Table 1. Also the results of the Monte Carlo simulations in our algorithm varied strongly for the different sensor models. In every Monte Carlo simulation, we performed 1,000 simulated runs of the robot and calculated the proportion of time steps in which the deviation of the robot from its trajectory exceeded $d_{\max}$. This proportion yields an estimate p_{MC} of $p_{\min}$ for the non-linear models. For all trajectories and all sensor models, the values of p_{MC} for the landmark sets our approach computed are stated in the fifth row of Table 1.

For the range-only sensor, p_{MC} is considerably above the intended value of 99% in all tasks. For the range-and-bearing sensor, p_{MC} is slightly below 99% in the pick-and-place task and for the bearing-only sensor, p_{MC} is below 99% in three of the five tasks. These results indicate that the non-linear components of the range measurements are less critical for landmark placement than those of the bearing measurements.

7.2.2. *Comparison to other Landmark Placement Strategies*

For comparison, we evaluated three other methods for placing landmarks in a way that the deviation guarantee is satisfied. Each method starts with a minimum number of landmarks and successively increases the number (or density) of landmarks until it finds a set for which the guarantee in the linearized system holds.

- *On trajectory* places landmarks equidistant on the desired trajectory.
- *On grid* places a landmark in the center of each cell of a regular grid. Starting with one cell covering the whole environment, the cell size is decreased at every iteration until the deviation guarantee holds. For efficiency, landmark positions which are outside the field of view of all states $\mathbf{x}^{\star}_{0:T}$ on the desired trajectory are not used.
- *Random* successively places landmarks at randomly chosen positions observable from the desired trajectory.

The number of selected landmarks and the values of p_{MC} for all landmark placement strategies are stated in Table 1. Dashes in the table indicate that no valid landmark configuration could be found. For all experiments, our approach placed fewer landmarks than the other approaches. The *on trajectory* method is the best method after ours for the *range-and-bearing* sensor, measured

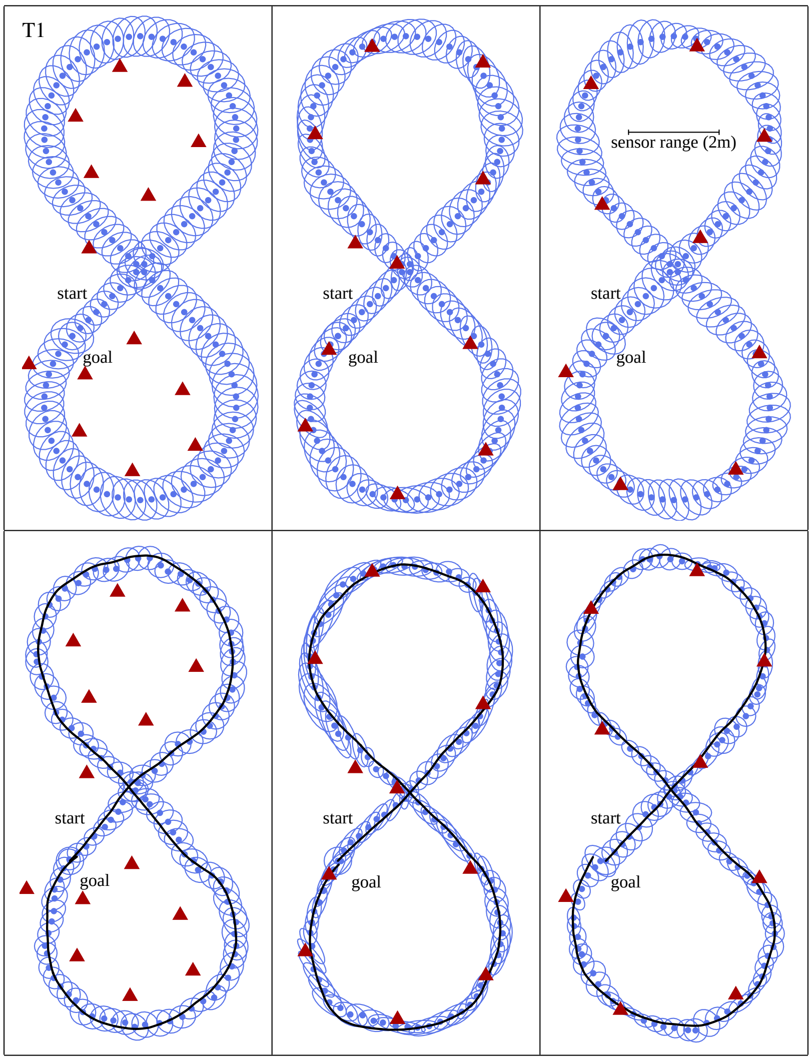

Figure 1: The landmark configurations (red triangles) our algorithm computed for the figure-eight trajectory T1 for three different sensor models: range-only (left), bearing-only (middle) and range-and-bearing (right). The blue points and ellipses in the upper row correspond to the means and the 99% covariance ellipses of the a priori distributions, and in the lower row to the a posteriori distributions of simulated sample runs. The true positions of the robot in the sample runs are depicted as black lines.

Table 1: Numbers of selected landmarks and results of Monte Carlo simulations

	Range-only sensor					Bearing-only sensor					Range-and-bearing sensor				
	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
Number of landmarks															
Our approach	14	12	11	18	10	11	9	7	16	7	9	8	6	13	5
On trajectory	–	–	25	–	58	41	–	12	–	23	12	9	10	17	7
On grid	48	32	38	56	23	26	19	17	30	18	20	20	17	25	16
Random	108	63	62	138	62	75	66	51	88	31	38	29	38	37	15
p_{MC}															
Our approach	0.999	0.998	0.999	0.999	0.999	0.979	0.978	0.991	0.994	0.826	0.999	0.997	0.998	0.994	0.986
On trajectory	–	–	0.996	–	0.962	0.353	–	0.955	–	0.773	0.996	0.999	0.983	0.999	0.980
On grid	0.999	0.999	0.999	0.999	0.998	0.997	0.981	0.995	0.999	0.931	0.996	0.999	0.995	0.999	0.999
Random	0.999	0.999	0.999	0.999	0.998	0.996	0.996	0.999	0.999	0.999	0.999	0.999	0.996	0.999	0.999

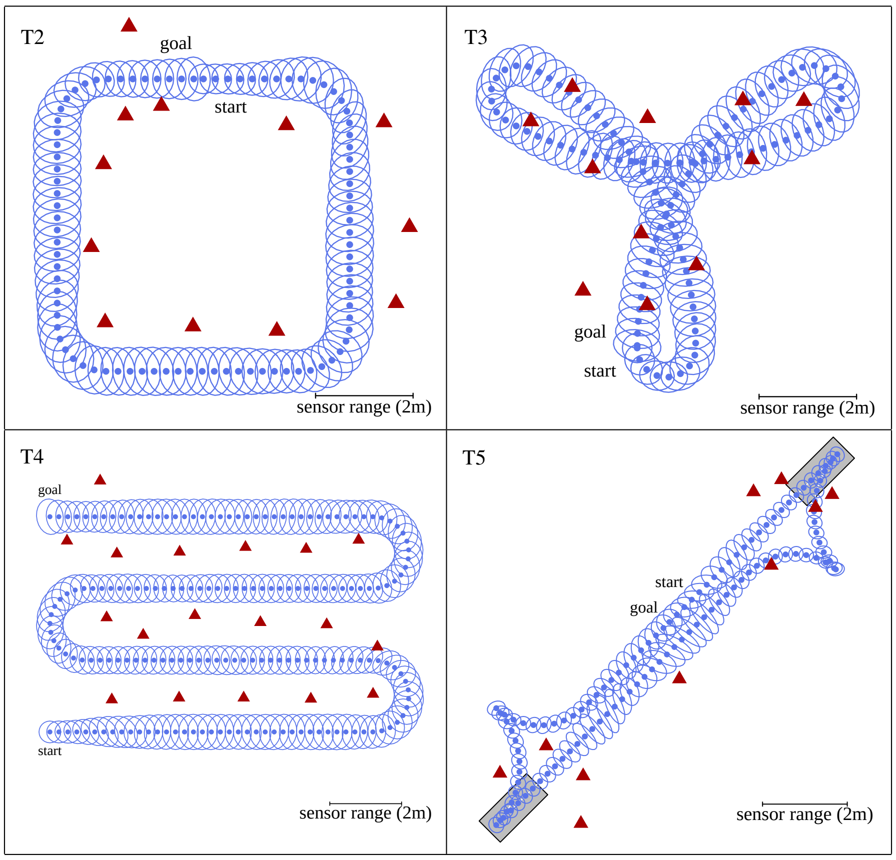

Figure 2: The landmark configurations (red triangles) our algorithm computed for four sample trajectories T2-T5 using a range-only sensor. T2 is a square, T3 a curved shape, T4 a sweeping trajectory, and T5 a *pick-and-place* task. The blue points and ellipses correspond to the means and the 99% covariance ellipses of the a priori distributions. In T5, the pick-up zone and the deposit zone are marked as gray rectangles.

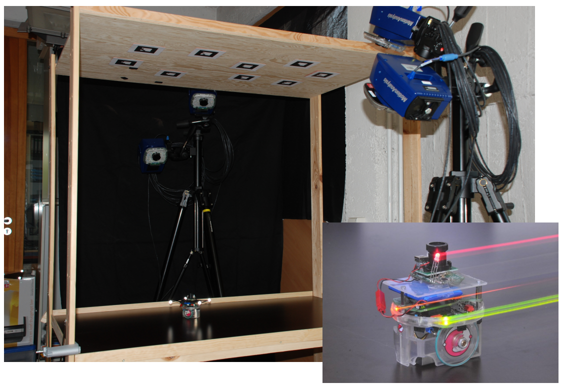
Figure 3: The miniature e-puck robot and its experimental environment. Mounted on top of the robot is a wireless webcam detecting the uniquely identifiable visual markers attached to the ceiling. The reference positions of the robot are obtained by the four-camera motion capture system.

by the number of landmarks placed. However, for the other two sensor models, the *on trajectory* method was not always able to find a landmark configuration which satisfied the guarantee in the linearized system. For this method, especially the non-linearities in the *bearing-only* sensor model resulted in low values for p_{MC}.

7.2.3. *Experiments with a Miniature Robot*

To further validate the simulation experiments for the scenarios in free space, we evaluated one of the landmark sets our algorithm generated also on the real e-puck robot [17] depicted in Fig. 3. As a range-and-bearing sensor, we utilized a webcam pointing upwards detecting uniquely identifiable ARToolkit markers [6] attached to the ceiling. We considered the navigation task T1 scaled down to suit the miniature size of our robot (diameter 75 mm) and the lower ceiling. Scaling the task by the factor 0.08 yields $d_{\max} = 0.04\,\mathrm{m}$. To evaluate the deviation $d^{\mathrm{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star)$ of the e-puck robot from its desired trajectory, we obtained the reference positions $\mathbf{x}_t$ from a MotionAnalysis motion capture system with four digital Raptor-E cameras. During 20 autonomous runs, $d^{\mathrm{pos}}(\mathbf{x}_t, \mathbf{x}_t^\star)$ was below $d_{\max}$ in 99.7% of the time steps.

7.3. *Placement in Structured Environments*

In the second set of experiments, we considered structured environments with walls and other obstacles at known positions. To represent the walls and obstacles, we used sets of lines, so-called line maps. In these experiments, we evaluated our algorithm for both spaces of possible landmark positions, the surfaces of the walls and obstacles $\mathcal{L}_{\mathrm{walls}}$ and the plane $\mathcal{L}_{\mathrm{free}}$. In the landmark placement algorithm, we used the sigma-point approach

Table 2: Comparison of Placement Spaces

		$\mathcal{L}_{\mathrm{walls}}$	$\mathcal{L}_{\mathrm{free}}$
T6	Length of Trajectory	218.5 m	
	Runtime	4 h 6 min	39 h 54 min
	Number of Landmarks	48	37
T7	Length of Trajectory	100.8 m	
	Runtime	13 min	5 h 50 min
	Number of Landmarks	21	16

described in Section 4.3.2 to check for landmark visibility. This approach allows us to take into account occlusions from walls when checking for landmark visibility, and it also allows us to consider non-circular fields of view of the robot. We simulated two navigation tasks in structured environments: T6 and T7 (see Fig. 4).

The line maps in both tasks were manually extracted from the grid maps shown in light gray in the figures. The map in the task T6 corresponds to the Willow Garage building (grid map recorded by Brian Gerkey) and the map in T7 corresponds to building 079 on the Freiburg University campus. For these tasks we simulated a differential drive robot equipped with a range-and-bearing sensor having a maximum range of 5 m and a half-circular field of view in front of the robot. Just as in the experiments in free space, we set $p_{\min} = 99\%$ and $d_{\max} = 0.5\,\mathrm{m}$.

7.3.1. *Comparison between Placement on Walls and Placement in Free Space*

We compared the runtime and the number of placed landmarks of our algorithm for the landmark placement in $\mathcal{L}_{\mathrm{walls}}$ to that of the placement in $\mathcal{L}_{\mathrm{free}}$ on tasks T6 and T7. For the landmark placement in $\mathcal{L}_{\mathrm{free}}$, we used an insight gained from empirical evaluations to speed up the computation. In the experiments, Algorithm 1 typically selected landmark positions $\boldsymbol{\ell}_{\mathrm{new}}$ inside the field of view of the current state of the robot $\mathbf{x}_\tau^\star$. Therefore, we restricted the search space for the optimizations to this area. For all evaluated tasks in our experiments, Algorithm 1 with the full search space did not select fewer landmarks than our implementation with the restricted search spaces. In structured environments, a similar restriction of $\mathcal{L}_{\mathrm{walls}}$ was not applicable, as in our experiments the field of view of $\mathbf{x}_\tau^\star$ often did not contain the best landmark position $\boldsymbol{\ell}_{\mathrm{new}}$. In fact, at some time steps it did not contain any possible landmark positions at all. Hence, we restricted the search space in the optimizations to all walls that are observable from the trajectory up to τ.

For both placement methods, in $\mathcal{L}_{\mathrm{walls}}$ and in $\mathcal{L}_{\mathrm{free}}$,

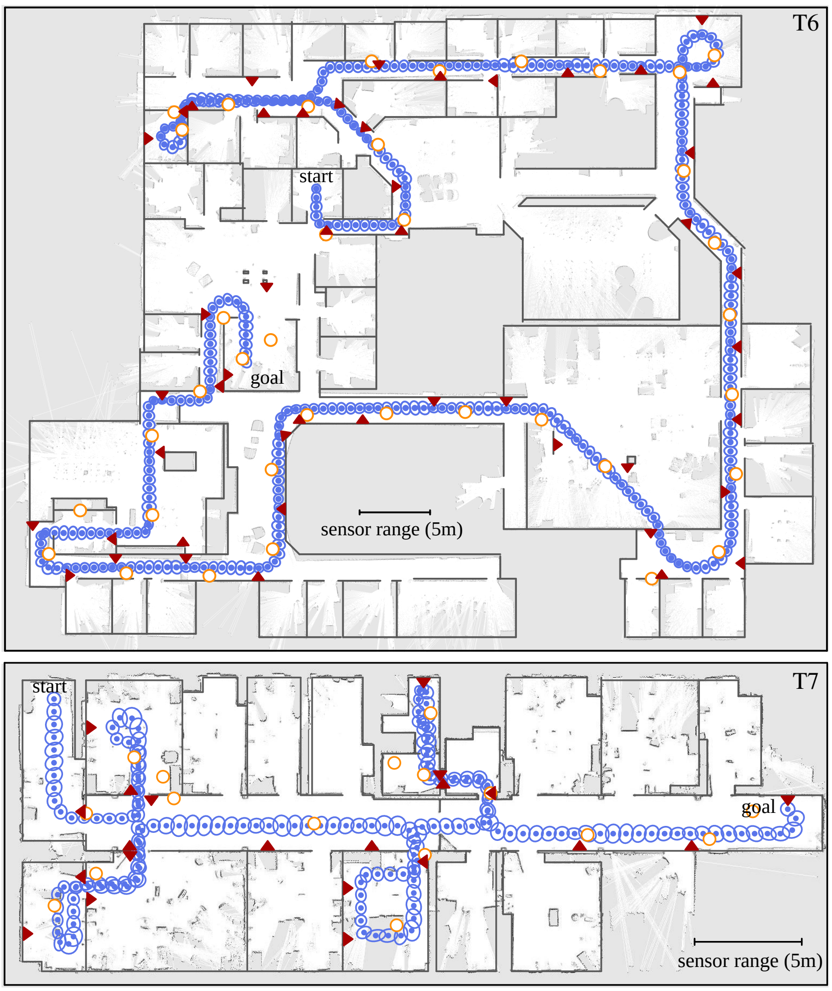

Figure 4: The landmark configurations our algorithm computed when placing landmarks on the walls of buildings (red triangles) and in free space (orange circles). The blue points and ellipses correspond to the means and the 99% covariance ellipses of the a priori distributions computed using the landmarks placed on the walls. The line maps of the buildings are depicted in dark gray, and the grid maps from which the line maps were extracted are shown in light gray.

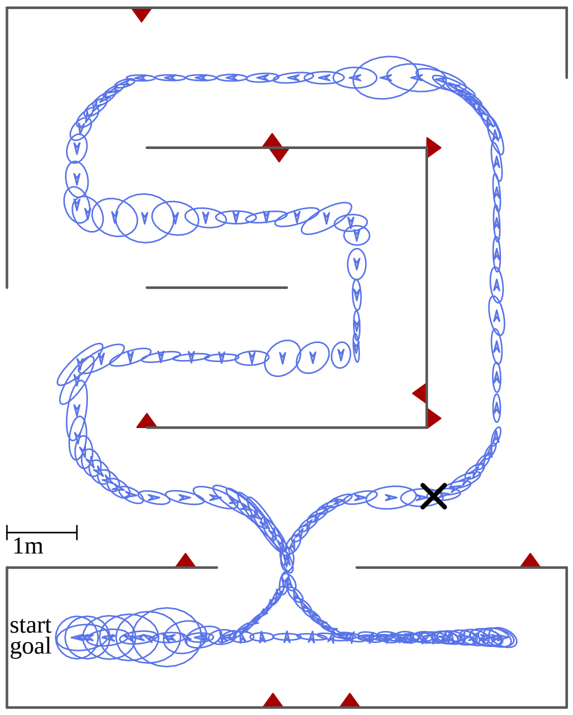

Figure 5: The landmark configuration (red triangles) our algorithm placed on the walls of the experimental environment of the holonomic robot. The blue arrows and ellipses correspond to the means and the 99% covariance ellipses of the a priori distributions. The line map of the experimental environment is shown in dark gray. The black cross marks the spot on which the robot is standing in Fig. 6.

we executed our algorithm single-threaded on an Intel® Core™ i7 2.8GHz with 12GB RAM. For T6 and T7, the lengths of the trajectories, the numbers of selected landmarks and the runtime of Algorithm 1 are listed in Table 2. As can be seen in the table, landmark placement on the plane $\mathcal{L}_{\text{free}}$ managed to fulfill guarantee (5) with fewer landmarks than landmark placement on the walls $\mathcal{L}_{\text{walls}}$. Hence, the landmark positions on the walls appear to be suboptimal. On the other hand, the runtime of the landmark placement in $\mathcal{L}_{\text{free}}$ is considerably longer than the runtime of the landmark placement in $\mathcal{L}_{\text{walls}}$. This is due to the fact that the search space $\mathcal{L}_{\text{free}}$ is considerably larger than $\mathcal{L}_{\text{walls}}$. However, selecting the space of landmark positions cannot be decided by considering these numbers but has to be decided depending on the physical properties of the utilized sensor and the landmarks.

7.3.2. *Long Term Experiments with a Holonomic Robot*

We evaluated the continuous operation property described in Section 5.3 in a long-term experiment with a real holonomic robot. The environment and the round-trip trajectory of this navigation task are shown in Fig. 5.

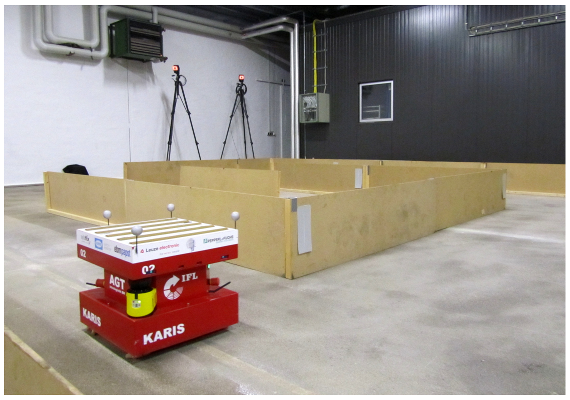

Figure 6: The holonomic KARIS robot operating in its experimental environment. It is equipped with two laser range finders mounted oppositely on the base for a 360° field of view. They detect reflective markers which are attached to the walls and are used as landmarks for localization. The reference positions of the robot are obtained by the motion capture system.

As robotic system, we used the KARIS robot [14] which is depicted in Fig. 6. This robot is designed as a logistics robot for industrial application in storage facilities and production sites. It is equipped with Mecanum wheels, which allow for holonomic motion. Two SICK S300 laser range finders are mounted on opposite sides of the robot. They return range measurements and intensity values in a 360° field of view with a resolution of 0.5°. Based on the intensity values, the robot can detect landmarks made of reflective tape mounted on walls. Experiments showed that these reflective markers are reliably observable by the robot only if the angle between the landmark orientation and the robot position is at least 22°. For landmark placement, we therefore used a sensor model with that observability constraint and with a circular field of view.

The number and positions of the landmarks selected by our algorithm highly depend on the covariance matrices M_t and N_t of the noise in the motion model and the sensor model, respectively. Therefore, before placing the landmarks, we did a calibration run with the robot, in which the pose of the robot was determined precisely by a MotionAnalysis motion capture system with nine digital Raptor-E cameras. We estimated the noise matrices as the maximum likelihood values with respect to the reference positions and the odometry or landmark measurements, respectively.

Taking into account the calibrated noise values, Algorithm 1 placed 11 landmarks that can be seen in Fig. 5. Using the observations of these landmarks for localiza-

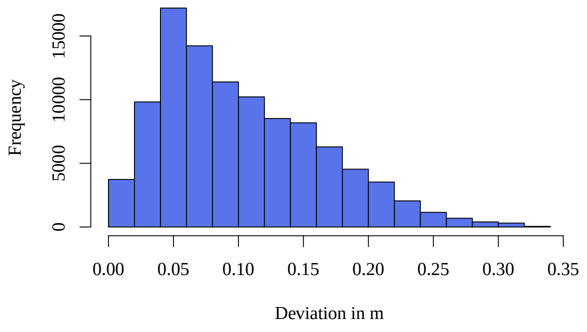

Figure 7: The histogram of the deviation of the KARIS robot from its desired trajectory during operation on a round trip for about three hours.

tion, the KARIS robot performed a continuous round trip on the defined trajectory for about 3 hours, corresponding to one battery charge. During this time, the robot completed the navigation task 62 times. The deviation of the robot from its desired trajectory was monitored at a 10 Hz rate by the MotionAnalysis motion capture system. The captured deviation values of the robot never exceeded $d_{\text{max}} = 0.5$ m for all 102,300 time steps. Fig. 7 shows the deviations from the trajectory in a histogram plot. As can be seen in the plot, the deviation of the robot from its desired trajectory was typically around 0.05 m. The largest deviation value measured was 0.337 m.

These experiments demonstrate that our approach is suitable for efficient placement of landmarks in unstructured or structured environments. The selected landmark configurations were proven to allow for a reliable navigation in extensive simulation and real world experiments with various robot platforms and sensing technologies.

8. Conclusions

In this paper, we have presented a landmark placement method that with high confidence guarantees a bound on the maximum deviation of the robot from its planned trajectory. In the landmark placement phase, our approach approximates the real motion and sensor models by their linearizations to efficiently evaluate the guarantee. In the subsequent validation stage, we apply a Monte Carlo simulation using the real system dynamics to check if the selected landmark set satisfies the deviation guarantee also for the possibly nonlinear models. In contrast to other approaches, our algorithm is customizable to specific robotic systems and navigation tasks and inherently chooses the appropriate number of landmarks needed. In extensive experiments, we demonstrated that our method outperforms other approaches. Furthermore, our algorithm was successfully applied to create landmark configurations for several simulated and real-world navigation tasks in which common robot platforms navigated reliably in long-term experiments.

Acknowledgments

This work has partly been supported by the German Research Foundation (DFG) within the Research Training Group 1103. The authors would like to thank Axel Rottmann for his implementation of the Powell algorithm and Christoph Sprunk for discussions and help concerning the holonomic motion model.

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. Near-optimal landmark selection for mobile robot navigation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[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, 2nd edition, 2000.

[6] M. Billinghurst and H. Kato. Collaborative augmented reality. *Communications of the ACM*, 45(7):64–70, 2002.

[7] A. Doucet, N. de Freitas, and N. Gordan. *Sequential Monte-Carlo Methods in Practice*. Springer Verlag, 2001.

[8] H.F. Durrant-Whyte, D. Pagac, B. Rogers, M. Stevens, and G. Nelmes. Field and service applications - an autonomous straddle carrier for movement of shipping containers. *IEEE Robotics & Automation Magazine*, 14:14–23, 2007.

[9] L. Erickson and S. LaValle. An art gallery approach to ensuring that landmarks are distinguishable. In *Proc. of Robotics: Science and Systems (RSS)*, 2011.

[10] J.-S. Gutmann, E. Eade, P. Fong, and M. Munich. A constant-time algorithm for vector field SLAM using an exactly sparse extended information filter. In *Proc. of Robotics: Science and Systems (RSS)*, 2010.

[11] D.B. Jourdan and N. Roy. Optimal sensor placement for agent localization. *ACM Trans. Sen. Netw.*, 4(3):1–40, 2008.

[12] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proc. of the American Control Conference*, 1995.

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

[14] A. Kleiner, D. Sun, and D. Meyer-Delius. ARMO - adaptive road map optimization for large robot teams. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[15] R. Lerner, E. Rivlin, and I. Shimshoni. Landmark selection for task-oriented navigation. *IEEE Transactions on Robotics*, 23(3):494–505, 2007.

[16] D. Meyer-Delius, M. Beinhofer, A. Kleiner, and W. Burgard. Using artificial landmarks to reduce the ambiguity in the environment of a mobile robot. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[17] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli. The e-puck, a robot designed for education in engineering. In *Proc. of the Conf. on Autonomous Robot Systems and Competitions*, 2009.

[18] M. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[19] C. Pradalier and S. Sekhavat. "Localization Space": a framework for localization and planning, for systems using a sensor/landmarks module. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2002.

[20] T. Rupp and P. Levi. Optimized landmark arrangement for absolute localization - a practical approach. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2000.

[21] P. Sala, R. Sim, A. Shokoufandeh, and S. Dickinson. Landmark selection for vision-based navigation. *IEEE Transactions on Robotics and Automation*, 22(2):334–349, 2006.

[22] J. Salas and J. Gordillo. Placing artificial visual landmarks in a mobile robot workspace. In *Proc. of the Ibero-American Conf. on Artificial Intelligence (IBERAMIA)*, 1998.

[23] H. Strasdat, C. Stachniss, and W. Burgard. Which landmark is useful? Learning selection policies for navigation in unknown environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[24] S. Thrun. Finding landmarks for mobile robot navigation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1998.

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

[26] S. Zhang, L. Xie, and M.D. Adams. Entropy based feature selection scheme for real time simultaneous localization and map building. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.