Learning Landmark Selection Policies for Mapping Unknown Environments

Hauke Strasdat Cyrill Stachniss Wolfram Burgard

H. Strasdat is with the Dept. of Computing, Imperial College London, SW7 2AZ, UK
C. Stachniss and W. Burgard are with the University of Freiburg, Dept. of Computer Science, D-79110 Freiburg, Germany

Summary. In general, a mobile robot that operates in unknown environments has to maintain a map and has to determine its own location given the map. This introduces significant computational and memory constraints for most autonomous systems, especially for lightweight robots such as humanoids or flying vehicles. In this paper, we present a universal approach for learning a landmark selection policy that allows a robot to discard landmarks that are not valuable for its current navigation task. This enables the robot to reduce the computational burden and to carry out its task more efficiently by maintaining only the important landmarks. Our approach applies an unscented Kalman filter for addressing the simultaneous localization and mapping problem and uses Monte-Carlo reinforcement learning to obtain the selection policy. In addition to that, we present a technique to compress learned policies without introducing a performance loss. In this way, our approach becomes applicable on systems with constrained memory resources. Based on real world and simulation experiments, we show that the learned policies allow for efficient robot navigation and outperform handcrafted strategies. We furthermore demonstrate that the learned policies are not only usable in a specific scenario but can also be generalized towards environments with varying properties.

1 Introduction

In recent years, there has been a trend towards embedded systems in robotics. A series of such approaches deal with autonomous cars, helicopters, blimps, underwater vehicles, and wheeled or humanoid robots. As embedded systems typically have much higher limitations with respect to the computational power and memory capacity, it is important in the context of embedded systems to develop efficient algorithms that scale with the computational constraints of the underlying hardware.

In robotics, one of the core capabilities needed for the majority of applications is autonomous navigation. For truly autonomous navigation in initially unknown environments, the robot has to solve the so-called simultaneous localization and mapping (SLAM) problem [2, 14, 24, 21]. Solving the SLAM problem, however, is computationally demanding and the memory requirements increase with the number of landmarks that need to be maintained by the robot. In practice, there are many scenarios in which the number of visible landmarks during a navigation task is significantly larger than the number of landmarks which can be processed efficiently using an embedded device. This leads to the question which landmark should be stored and maintained by the robot to optimally solve the navigation task. A landmark is only useful if it contributes to keep an accurate pose estimate of the robot at the right time and in such a way that it is valuable for the navigation task. In this paper, we present an approach for learning a landmark selection policy that optimizes the navigation task carried out by the robot given its computational or memory constraints. It is obvious that the utility of a landmark depends on the type of navigation task. We analyze two types of navigation tasks: A single-goal navigation task and a round-trip navigation task where subgoals are visited more then once. One major advantage of our approach is that the policies are not limited to the environment they have been learned in. Rather, they can also be applied successfully in environments with different properties of the underlying landmark distribution. Furthermore, we presented a way to compress learned policies so they become applicable on systems with reduced memory resources.

This paper is organized as follows. After a discussion of related work, Section 3 briefly introduces the unscented Kalman filter and its application to SLAM as well as reinforcement learning. Section 4 then describes the different navigation tasks considered in this paper. After that, we introduce our approach to learn the optimal landmark selection policy. Finally, we present experimental results carried out in simulation as well as on a real wheeled robot.

2 Related Work

The standard method for SLAM relies on the extended Kalman filter (EKF) [12] or its variants such as the unscented Kalman filter (UKF) [8]. Using these approaches, the computational requirement and memory demand increase at least quadratically with the number of landmarks since the full correlation between the position of all landmarks is taken into account. There are many approximative filtering techniques for SLAM [14, 24]. These methods do not incorporate the full correlation between the landmarks, so that the computational constraints are less restrictive. However, their memory demand increases at least linearly with the number of landmarks used.

Recently, Sala *et al.* [18] presented a graph-theoretic formulation for the selection problem of visual features to perform navigation in known environments. The optimal set of features is defined as the minimal set with which navigation is possible. Zhang *et al.* [26] proposed an entropy-based landmark selection method for SLAM. This method specifies a measure of which visible landmark is best in terms of entropy reduction. However, it only provides a vague guideline for how many features should be selected at a given point in time. Furthermore, Lerner *et al.* [11] presented another quality measure for landmark selection in known environments which is based on the comparison of pose uncertainties. Dissanayake *et al.* [6] suggested a map management which ensures a uniform distribution of landmarks over the traversed area. Apart from landmark selection, other active methods were presented such as maximizing the SLAM estimate by intelligent path planning [5], or increasing the performance of a soccer playing robot by active sensing [10]. Recently, Hornung *et al.* [7] proposed a system for learning acceleration policies in the context of vision-based navigation. Furthermore, they presented a technique to compress the learned policy using a clustering approach. Several other policy/state space compression techniques for reinforcement learning were presented in the past [25][13]. While these techniques mainly focus on gaining a speed-up during learning, our policy compression approach is similar to Hornung *et al.*'s [7] and motivated by the storage problem: How can we represent the learned policy in a most compact way so that it becomes applicable on memory-constrained systems and so that, at the same time, the compression does not lead to a loss of performance?

In this paper, we present a universal approach for landmark selection in unknown environments. The value of a landmark is measured in terms of how well it improves the navigation/localization capabilities of the robot given the targeted navigation task. This is especially important for robots with restricted resources. We learn a landmark selection policy using Monte-Carlo reinforcement learning [3, 20, 23] and k-nearest neighbor regression [19]. We show in real world and simulation experiments that this technique allows for more efficient robot navigation. Furthermore, we demonstrate how the learned policies, which consist of tens of thousands of parameters, can be compressed using neural networks without a loss of performance. In this way, our approach becomes applicable on memory-constrained systems and extends our previous work [22].

3 Preliminaries

3.1 Unscented Kalman Filter

The unscented Kalman filter (UKF) is a recursive Bayes' filter that estimates the state $\mathbf{x}$ of an dynamical system in discrete time steps given a sequence of actions $\mathbf{u}$ and observations $\mathbf{z}$. The n-dimensional state vector $\mathbf{x}$ is represented by a multivariate Normal distribution with mean μ and covariance matrix $\Sigma^{(n\times n)}$. The dynamics of the system are described by a *state transition function* g plus Gaussian noise $\epsilon_{g,t}$: $\mathbf{x}_t = g(\mathbf{u}_t, \mathbf{x}_{t-1}) + \epsilon_{g,t}$. Measurements are integrated using the *observation function* h: $\hat{\mathbf{z}}_t = h(\mathbf{x}_t) + \epsilon_{h,t}$. Again, Gaussian noise $\epsilon_{h,t}$ is added. Since Kalman filtering is an approach for systems governed by a linear difference equation, special efforts must be made to take the non-linearities in g and h into account.

The key idea of the unscented Kalman filter, which has been introduced by Julier and Uhlmann [8], is to apply a deterministic sampling technique that is known as the unscented transform to select a small set of so-called sigma points around the mean. Then, the sigma points are propagated through the non-linear functions. Afterwards, mean and covariance estimates are computed based on the transformed points. The advantage of this technique is that the filter can much better deal with non-linearities and thus lead to a more robust technique than the EKF.

3.2 Simultaneous Localization and Mapping

In the context of the SLAM problem, one seeks to simultaneously determine the map of the environment and the pose of the robot. Probabilistic methods seek to estimate the joint probability distribution

$$p(\mathbf{p}_t, \mathbf{l}_1, \ldots, \mathbf{l}_M \mid \mathbf{u}_1, \ldots, \mathbf{u}_t, \mathbf{z}_1, \ldots, \mathbf{z}_t) \tag{1}$$

about the pose $\mathbf{p}_t$ of the robot at time t and the position of the landmarks $\mathbf{l}_1, \ldots, \mathbf{l}_M$ given all previous motions $\mathbf{u}_1, \ldots, \mathbf{u}_t$ and observations $\mathbf{z}_1, \ldots, \mathbf{z}_t$. Various approaches to estimate this posterior have been presented in the literature.

In this paper, we address the SLAM problem using the UKF by representing the joint state $(\mathbf{p}_t^T, \mathbf{l}_1^T, \ldots, \mathbf{l}_M^T)^T$ with $\langle \mu, \Sigma \rangle$. This is a standard approach which has been shown to operate successfully in the past. For convenience, we abbreviate the mean of the robot pose $(\mu_1, \mu_2, \mu_3)^T$ as $(x, y, \theta)^T$. The mean of the jth landmark location $(\mu_{2j+2}, \mu_{2j+3})^T$ is denoted by $\left(l_x^{[j]}, l_y^{[j]}\right)^T$. Furthermore, we interpret the state transition function h as the robots motion model. In addition, we assume that range and bearing observations $(\rho, \phi)^T$ are given so that we can define a corresponding observation model g. In our work, we initialize new landmarks in a single step as described in [2].

Note that our landmark selection framework is not limited to UKF or Kalman filter-based approaches and any other method can be applied for addressing the SLAM problem.

3.3 Monte-Carlo Reinforcement Learning

The basic idea of reinforcement learning [23] is to learn by the interaction with the environment. We consider a dynamic system consisting of an agent and its environment at discrete time steps τ. At each point in time τ, the world is in state $s_\tau \in \mathcal{S}$ and the agent chooses an action $a \in \mathcal{A}$. Then, the world transform into a new state $s_{\tau+1}$ and the agent receives a reward $r_{\tau+1} \in \mathcal{R}$. The goal is to maximize the return

$$R_\tau = \sum_{k=\tau+1}^{\mathcal{T}} r_k, \tag{2}$$

whereas $\mathcal{T}$ is the total number of time steps of one learning episode. The agent is following a policy

$$\pi(s, a) := p(a \mid s) \quad \forall s \in \mathcal{S} \tag{3}$$

which represents the probability of choosing action a under the assumption of being in state s. Each policy π has a corresponding Q-function

$$Q^\pi(s, a) := E_\pi\{R_\tau \mid s_\tau = s, a_\tau = a\} \tag{4}$$

which specifies the expected return R from choosing action a in state s. During the learning process, we would like to approximate the optimal policy $\pi^*(s, a)$ that

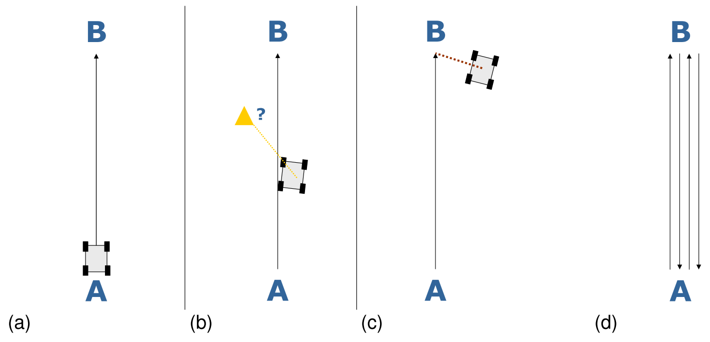


Fig. 1. Illustration of the single-goal navigation task (a-c) and the round-trip task (d).

maximizes the expected return. Therefore, we have to approximate the corresponding Q-function simultaneously.

One way of solving the reinforcement learning problem is based on Monte Carlo methods [3, 20]. Here, we estimate the Q-function as the average return over sample episodes. Initially, the Q-function is initialized with a prior R_{prior}. During the training, a *soft policy* should be used. Thus, it should hold that $\pi(s, a) > 0$ for all possible state-action pairs in order to assure that each state is reachable during the training process. One common soft policy is ϵ-greedy which selects with the high probability of $1 - \epsilon$ the action

$$a^* = \arg\max Q(s, a) \tag{5}$$

that maximizes the expected return and a random action otherwise.

Note that the time index t used in the SLAM setting is not necessarily identical with the discrete time at which the reinforcement learning framework has to make decisions. Therefore, we introduced a second time index τ.

4 Navigation Tasks

4.1 Single-goal Task

Let us consider the following most basic navigation task (see Fig. 1 (a-c)). The robot is located at position A. It is supposed to drive from there to the goal position B. In this example, the robot's motion is affected by a drift. In addition, N landmarks are distributed randomly over the environment. When the robot perceives a new landmark, it has to decide whether it should integrate this landmark in the UKF or not. The UKF has a landmark capacity of M landmarks with $M << N$. The goal is to choose the landmarks in such a way that the distance of the final position of the robot $(x_T, y_T)^T_{\text{true}}$ and the target position B is minimized. Hence, we define the reward

$$r_\tau = \begin{cases} -\left|B - (x_T, y_T)^T_{\text{true}}\right| & \text{if } \tau = \mathcal{T} \\ 0 & \text{else,} \end{cases} \tag{6}$$

as the negative Euclidean distance of the robot's true position to the goal B if the training episode reaches the terminal state $s_{\mathcal{T}}$; intermediate rewards $r_1,\ldots,r_{\tau-1}$ are set to zero.

4.2 Round-trip Task

In the round-trip task, the robot is supposed to reach several subgoals (see Fig. 1 (d)). First, it starts at A and it is supposed to drive to B, then back to A twice. A new subgoal is selected as soon as the position estimate of the robot $(x_t, y_t)^T$ is close to it – independent of the robot's true position $(x_t, y_t)^T_{\text{true}}$. In this task, the error in the pose estimate should be minimized over the whole trajectory. For convenience, we specify the return directly as the negative average error over the remaining trajectory,

$$R_\tau = -\frac{1}{|T - t(\tau)|} \sum_{t'=t(\tau)}^{T} \left| \begin{pmatrix} x_{t'} \\ y_{t'} \end{pmatrix}_{\text{true}} - \begin{pmatrix} x_{t'} \\ y_{t'} \end{pmatrix} \right|, \tag{7}$$

whereas $t(\tau)$ specifies the time when the τth decision is made and T is the time when the robot reaches its final destination. To simplify things for the second task, landmark selection is only allowed while the robot moves from A to B the first time. The round-trip task is more complex than the previous one. However, it is worth considering since it focuses on the loop-closing problem of SLAM and therefore has a higher practical relevance than the single-goal task.

5 Navigation and Landmark Selection

5.1 Motion Control

The robot is steered towards the subgoals using a straightforward controller. An appropriate translational acceleration $\dot{\omega}_t$ and rotational acceleration $\dot{\upsilon}_t$ is selected based on the current estimate of the robot pose $\mathbf{p}_t$, the translational velocity ω_t and rotational velocity υ_t.

5.2 Learning Landmark Selection Policies

In order to learn landmark selection policies with Monte Carlo reinforcement learning, we need to define the state space $\mathcal{S}$ and the action space $\mathcal{A}$. In addition to that, we need to find an appropriate representation for the continuous Q-function. But first, we would like to highlight what makes this problem difficult.

Challenges

Learning a landmark selection policy for navigation tasks is a hard problem. Due to the stochasticity of robot motion, optimal landmark selection can still lead to a

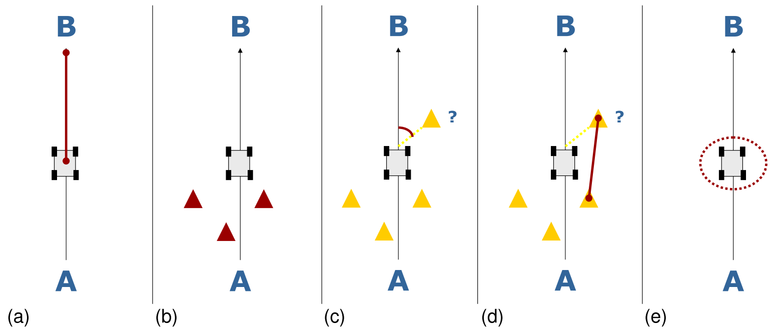


Fig. 2. Illustration of the state space.

significant localization error of the vehicle. At the same time, the robot may reach a desired target location accurately by chance even if no landmarks are selected at all. Thus, the training data used in our learning domain is comparably noisy. In the long run, however, the average localization error reflects to the quality of a specific selection strategy. Furthermore, it is crucial to remember that we perform navigation in *unknown environments*. Hence, it is not the goal to memorize a specific environment, but to learn general principles. For this reason, each learning episode is performed in a different environment. This makes the training data even more noisy: A selection policy which is good for one environment might not be a good policy in general (i.e., concerning the whole space of possible environments).

State Space

The available state information consists of the UKF state $\langle \mu, \Sigma \rangle$ and the current range and bearing observation $(\rho, \phi)^T$. This full information would lead to an high-dimensional state space so that successful learning is impractical. It is therefore desirable to reduce the space while preserving as much as possible of the relevant information. This can be achieved by defining features that summarize the essential information. One basic feature is the position of the potentially new landmark,

$$\begin{pmatrix} l_x^{[\text{new}]} \\ l_y^{[\text{new}]} \end{pmatrix} = \begin{pmatrix} x_t + \rho \cos(\phi + \theta_t) \\ y_t + \rho \sin(\phi + \theta_t) \end{pmatrix}, \tag{8}$$

according to the current range and bearing observation $(\rho, \phi)^T$ and the robot's pose estimate $(x_t, y_t, \theta_t)^T$. Additionally, we define the following five features:

1. Estimated distance to subgoal B (see Fig. 2 (a)),

$$d_{\text{est}} = \left| B - (x_t, y_t)^T \right|, \tag{9}$$

2. Number of landmarks integrated in the UKF (see Fig. 2 (b)),

$$m = \left| \{ j \in M : \Sigma_{2j+2} < \infty \wedge \Sigma_{2j+3} < \infty \} \right|, \tag{10}$$

where Σ_{2j+2} and Σ_{2j+3} are the variances of the jth landmark in the x and y direction.

3. Yaw angle to potential new landmarks ϕ (see Fig. 2 (c)),
4. Distance of the potentially new landmark to the closest landmark already integrated (see Fig. 2 (d)),

$$d_l = \min_{\substack{j \in L \\ \text{with } \Sigma_{2j+2} < \infty \wedge \Sigma_{2j+3} < \infty}} \left| \begin{pmatrix} l_x^{[j]} \\ l_y^{[j]} \end{pmatrix} - \begin{pmatrix} l_x^{[\text{new}]} \\ l_y^{[\text{new}]} \end{pmatrix} \right|, \qquad (11)$$

5. Uncertainty of the robot pose $\Sigma^{3\times 3}$ in terms of its entropy (see Fig. 2 (e)),

$$H = \ln(\sqrt{(2\pi e)^3 |\Sigma^{3\times 3}|}). \qquad (12)$$

The first of these features summarizes the robot position (x_t, y_t). The landmark positions $\mathbf{l}_1, \ldots, \mathbf{l}_M$ are summarized by the fourth feature while the new observation $(\rho, \phi)^T$ is represented by the third feature as well as fourth one. The covariance Σ_t is comprised by the second feature and the fifth one.

In the following, we will consider three different variants of the learning approaches. The first approach only relies on a two dimensional state space (first and second feature), the second one uses an four dimensional feature space (first to fourth feature) and the third one uses five dimensions (all five features).

Function Approximation

Since the state space of the features is continuous (with the exception of the second dimension), we need to estimate the Q-function with some function approximator. In our current implementation, we use k-nearest neighbor (k-NN) regression [19]. Training points – i.e. state/action values (s, a) which are each labeled with a return R – are efficiently stored in set of kd-trees [4, 1]. The jth kd-tree represents the returns from choosing an action a_j in a given state s. In the kd-tree representation, each dimension of the data points are normalized between zero and one, so that spherical search regions become practicable. If a query (s', a') is performed, the k nearest data points to the query point s' (w.r.t. Euclidean distance) are selected from the appropriate kd-tree. The return is estimated as unweighted average over the corresponding R-values. If less then k_{min} data points are found within a fixed radius around the query point, a prior R_{prior} is returned instead. In our current implementation, we set $k = 50$ to reflect the high amount of noise in our training data (see Sec. 5.2); k_{min} is set to 10. The k-NN regression approach has the advantage over the common grid-based discretization methods that it has a high degree of generalization in areas where the density is low and it is precise in regions where the data points are dense. In contrast to non-linear models such as neural networks [17], no over-fitting occurs. As opposed to other regression techniques, in which the model is also expressed directly in terms of their training data such as Gaussian Processes [15], k-NN regression is very fast. Even with hundreds of thousands of data points, a query can be performed in a few milliseconds. An efficient evaluation is essential for learning in practice, since the regression has to be carried out frequently. In various tests, we could not reveal a significant benefit from using Gaussian process regression over k-NN regression for reinforcement learning in our domain. Due to space restrictions, these experiments are omitted in the experimental section.

Action Selection

In our learning problem, the action is a binary decision:

$$\mathcal{A} = \{a_{\text{reject}}, a_{\text{accept}}\} \tag{13}$$

This means that either the potential new landmark is chosen or not. In order to boost the training, a variant of ϵ-greedy is used:

$$\pi(s) = \begin{cases} \arg\max Q(s,a) & \text{if } Q(s,a_{\text{reject}}) \neq Q(s,a_{\text{accept}}) \\ & \text{ and } \chi_1 < 1-\epsilon \\ a_{\text{accept}} & \text{if } [Q(s,a_{\text{reject}}) = Q(s,a_{\text{accept}}) \text{ or } \chi_1 \leq \epsilon] \\ & \text{ and } \chi_2 < \frac{M}{N_{\text{visible}}} \\ a_{\text{reject}} & \text{else} \end{cases} \tag{14}$$

Here, χ_1 and χ_2 are uniform random samples between zero and one; N_{visible} is the expected number of visible landmarks in one training episode. Using this soft policy in the beginning of the training – when $Q(s,a_{\text{reject}}) = Q(s,a_{\text{accept}}) = R_{\text{prior}}$ in most cases – landmarks are selected with the probability of $\frac{M}{N_{\text{visible}}}$. Thus, it is ensured that landmarks are selected over the whole trajectory. Neither landmarks in the beginning of the episode nor landmarks in the end are preferred. If standard ϵ-greedy is used, a_{accept} and a_{reject} would be chosen with a probability of 0.5 each. Hence, depending on the values for the landmark capacity M and expected number of visible landmarks N_{visible} it could happen that either all landmarks are selected in the beginning of the episode or that considerably fewer landmarks than M are selected. Either would lead to a slow convergence rate.

To sum up, we use a learning approach for landmark selection based on Monte-Carlo reinforcement learning and k-NN regression. The state space is compactly represented by five features and the action is a binary decision.

5.3 Generalization

Until now, we considered an approach to learn a selection policy in unknown environments, but for a specific scenario. However, it is desirable to train a policy in one scenario and then apply this policy in another setting. Important parameters of a training scenario are the number N of landmarks in the environment and the landmark capacity M of the UKF. To generalize, it is important to have a scenario-independent state space representation. For instance, instead of the number of landmark integrated in the UKF m, we need to speak about the percentage of landmarks $\frac{m}{M}$.

5.4 Policy Compression

Policies learned using k-NN-regression usually consists of tens of thousands of data points. To apply these policies on memory-constrained systems, we need to produce a compressed representation. One possibility is to compress the Q-function directly [25][13]. However, the Q-function also represents areas of the state/action space which are rarely visited when applying a greedy policy. Additionally, the Q-function maps each state/action value to a return value, whereas we are only interested in the action to apply given a state and not the specific return value. Therefore, we follow the approach of Hornung *et al.* [7] and perform the greedy policy n times (here, $n = 1,000$) and log the occurring decisions. As a result, we get a set of 10,000 state/action pairs. Since we only have two different actions, a_{reject} and a_{accept}, these pairs can be seen as labeled training data of a binary classification task. In order to compress the policy, any supervised classification technique can be applied which has a small number of model parameters. Here, we use a neural network [17] with one hidden layer, Rprop [16] for the weight optimization, and sigmoid activation function. This leads to a continuous output between zero and one, whereas output less than 0.5 are interpreted as zero (=a_{reject}) and otherwise as one (=a_{accept}).

6 Experiments

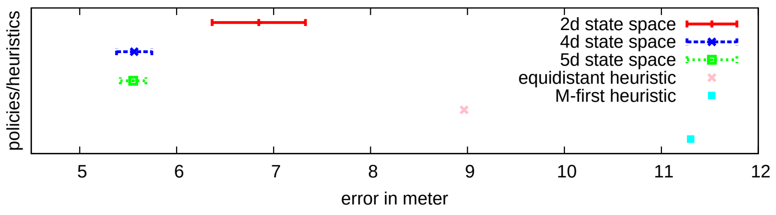


Fig. 3. Average performance of the trained policies and heuristics w.r.t. 1,000 test episodes. For the trained policies, the mean over the ten training runs as well as the corresponding 95%-confidence interval is shown.

6.1 Single-goal Task in Simulation

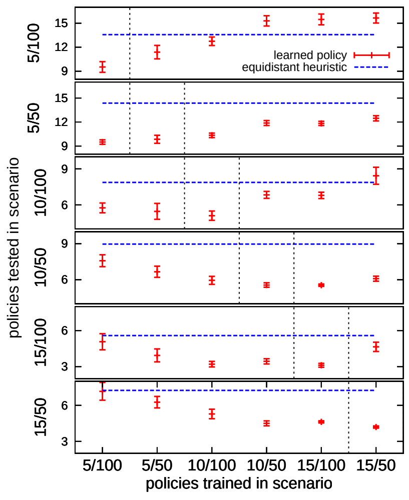
We evaluate the performance of our learning procedure for the single-goal task in a simulated environment. We choose an environment where N landmarks are randomly distributed in a $30m$ by $60m$ area. The distance between the start position A and the goal B is set to $44m$. We train our policy for 1,000 episodes. In each episode, landmarks are randomly re-distributed. We compare the trained policies with two heuristics. The first one is the *M-first heuristic* which simply integrates the M first landmarks that are observed. An apparently better policy is the *equidistant heuristic*. With this heuristic, the robot only integrates a new landmark after it has driven a certain distance so that the landmarks are approximately uniformly distributed over the whole trajectory (similar to [6]).

For the learning, we use k-NN regression with a two-, a four-, and a five-dimensional state space (see Section 5.2). At first, we consider an UKF with a landmark capacity of $M = 10$ and an environment with $N = 50$ landmarks. For each learning approach, ten training runs are performed. Each trained policy and heuristic is evaluated in 1,000 different environments (see Fig. 3). The one-sample t-test with a significance level of $\alpha = 0.05$ shows that all three learning approaches are significantly better than the equidistant heuristic. Furthermore, it is shown using a two-sample t-test that k-NN regression with a four dimensional state space leads to a significant smaller error than k-NN with two dimensions. Thus, the third feature, which is the distance d_l of a new landmark to landmarks already integrated, and the fourth one, which is the angle ϕ to the new landmark, seem to include relevant information which are not encoded in the first two dimension of the state space. Further experiments revealed that indeed both features are essential. However, we were not able to show that there is any benefit from including the fifth feature, the entropy H of the robot's pose. Even with a significance level of $\alpha = 0.25$, the t-test did not reveal a difference between the learning approach using the four-dimensional state space and the one using five dimensions.

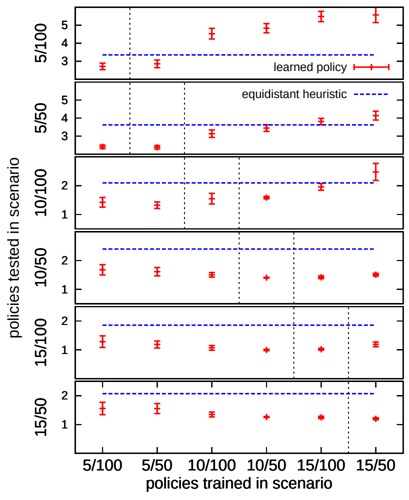
In order to evaluate how good the trained policies generalize, we trained and tested a policy in environments with $N = 50$ as well as $N = 100$ landmarks. In addition, we use UKFs with a capacity M of five, ten, and 15 landmarks. Fig. 4 (a) illustrates the high degree of generalization of our learning approach. For instance, if we perform a training in a setting with $N = 50$ and $M = 5$, we see that the trained policy leads to significantly better results than the equidistant heuristic in all six test scenarios. This indicates that our approach generalized over different landmark densities which is similar to environments of different scale and sensor range.

6.2 Single-goal Task Performed in a Real World Experiment

Furthermore, we evaluated our learning approach in a laboratory environment. Visual markers [9] have been randomly attached to the ceiling in a $2.5m$ by $5m$ area. Our used robot, a Pioneer 2DX-8, is equipped with an upward-looking camera and a SICK laser range scanner (see Fig. 5). The camera is used for the experiments observing landmarks at the ceiling whereas the laser is used for (near) ground truth evaluation. Since the odometry of the robot was too accurate in the limited space in which we carried out the experiment, we added a rotational bias of 0.1 rad per meter. It is impractical to train the policy in the real-world because this would not only require us to perform hundreds of training episodes but also to install different landmark distributions for each training episode. Thus, we trained the policy in simulation and tested it in the real-world setting. We also compared the trained policy to the equidistant heuristic. Both, the trained policy as well as the equidistant heuristic were tested ten times. The trained policy results in an error of 0.50 ± 0.08 whereas



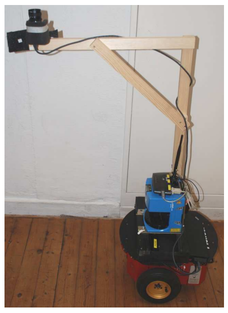
(a) Single-goal task



(b) Round-trip task

Fig. 4. High degree of generalization in the single-goal task (a) and the round trip task (b). The mean error over ten training runs and the corresponding standard derivation is shown. All policies below the dashed line are significantly better than the equidistant heuristic ($\alpha = 0.05$).

the equidistant heuristic leads to an error of 0.66 ± 0.07. Hence, the trained policy is significantly better than the equidistant heuristic (w.r.t. a t-test with $\alpha = 0.05$).



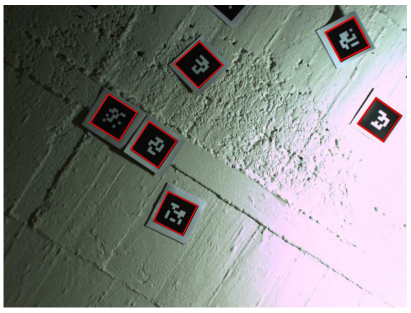


Fig. 5. Pioneer 2-DX8 robot with upward-looking camera and SICK laser range scanner (left) and detected visual landmarks on the ceiling (right).

6.3 Round-trip Task

The performance of our learning procedure for the round-trip task is evaluated in a simulated environment in a similar way to the single-goal task. It was trained using

k-NN regression with a four-dimensional state space over ten training runs. However, the error is defined as the average localization error over the whole trajectory. Again, we compare our learning with the equidistant heuristic. Fig. 4 (b) shows that the learned policy is significantly better than the heuristic. Furthermore, it is shown that we were able to generalize over the UKF capacity M as well as the number of landmarks N.

6.4 Policy Compression using Neural Network

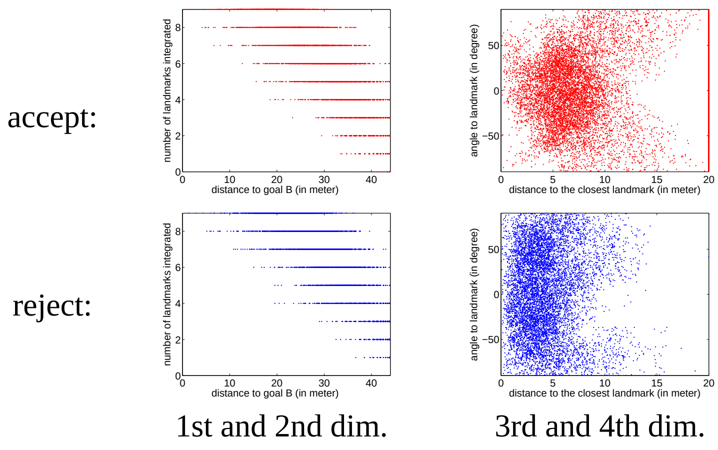
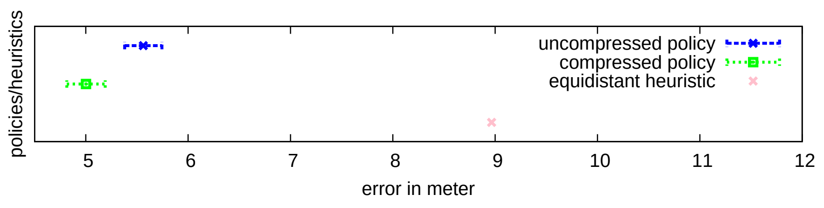


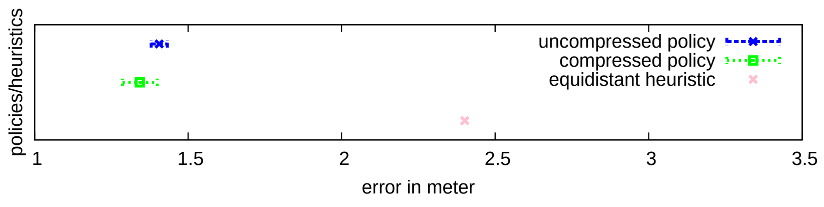
Fig. 6. Uncompressed strategy of the single-goal task. A projection of the four-dimensional state space on the first and second dimensions (left) as well as on the third and fourth dimension (right) is shown.

The uncompressed policy for the single goal task is illustrated in Fig. 6. The policy is represented using approximately 20.000 four-dimensional state points which are either labeled with a_{accept} or a_{reject} (see Sec. 5.4). The illustration indicates that there is a large overlap between the two classes. This is most likely due to the noisiness of the training data, i.e., due to the fact that the very same policy can lead to different returns. However, it is important to notice that only *projections* of the four-dimensional state space are shown and that the overlap is not that severe locally. Since the state space of the round-trip task looks similar, it is not shown here.

We compressed our learned policy for the single-goal task and the round-trip task using neural network classification (for M=5 and N=50). Since we have a four-dimensional state space and we want to learn a binary decision, we use a neural network with four input units and one output unit. It turned out that a hidden layer with three units is sufficient. This leads to a model with 19 parameters. To summarize, we were able to compress the learned policies which were roughly represented by 80,000 parameters (20,000 four-dimensional points) using a model with only 19 parameters.



Single-goal task



Round-trip task

Fig. 7. Comparison of the compressed and uncompressed strategy of the single-goal task (top) and the round-trip task (bottom). For the trained policies, the mean over the ten training runs as well as the corresponding 95%-confidence interval is shown.

For both navigation tasks, we learned policies in ten training runs. Each of the learned policies was compressed using the method describe above. Comparisons between the compressed and the uncompressed policies are shown in Fig. 7. One can see that there is no performance loss using the compressed policies in place of the uncompressed ones. Surprisingly, the compressed policy for the single-goal task is even significantly better than the uncompressed one. This could be explained by the generalization property of neural networks. To analyze this more precisely, it is worth looking at the compressed policy. Since the neural network parameters are hard to interpret, we apply the same approach as before. We apply the compressed policy in 1,000 different environments and log the occurring decisions. The resulting labeled data points are shown in Fig. 8. Although, the policy looks similar to the uncompressed one (Fig. 6), there are differences. For instance the neural network was able to better model the decisions boundaries. Because of its smoothness, it performed better on the unknown test data.

7 Conclusion and Future Work

In this paper, we presented an novel approach for landmark selection in unknown environments using reinforcement learning. The ability of a mobile robot to incorporate a landmark into its belief or to discard it allows for efficient robot navigation under computational constraints. The presented method is able to determine which landmark is valuable for the robot to efficiently solve its current navigation task.

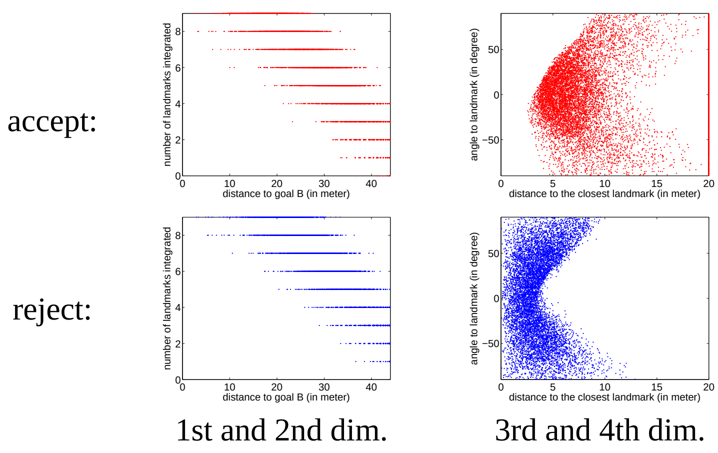


Fig. 8. Compressed strategy of the single-goal task. A projection of the four-dimensional state space on the first and second dimensions (left) as well as on the third and fourth dimension (right) is shown.

This is especially important for robots with restricted resources. We demonstrated by a series of real world and simulation experiments that the learned policies outperform handcrafted heuristics. Furthermore, we showed that a learned policy has a high degree of generalization since it can be applied in different environments with changed underlying parameters. Finally, we were able to compress the learned policies by means of neural network classification without introducing a performance loss.

Despite these encouraging results, there is space for further improvement. One interesting aspect is the possibility to delete an already incorporated landmark. It should be noted that this scenario is significantly more complex compared to the scenarios considered in the experimental section. For instance, the state space must be extended in order to represent the already integrated landmarks. In initial experiments, we figured out that good strategies keep a set of landmarks fixed for re-localization and perform an incremental pose correction with set of frequently replaced landmarks.

References

1. S. Arya and D.M. Mount. Algorithms for fast vector quantization. In *Proc. of the IEEE Data Compression Conference (DDC'93)*, pages 381–390, 1993.
2. T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, University of Sydney, 2002. pages 22–23.
3. A. Barto and M. Duff. Monte Carlo matrix inversion and reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 687–694, 1994.
4. J.L. Bentley. K-d trees for semidynamic point sets. In *Proc. of the 6th ACM Symposium on Computational Geometry*, pages 187–197, 1990.

5. M. Bryson and S. Sukkarieh. Active airborne localisation and exploration in unknown environments using inertial SLAM. In *Proc. of the IEEE Aerospace Conference*, 2006.
6. G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA'00)*, pages 1009–1014, 2000.
7. A. Hornung, H. Strasdat, M. Bennewitz, and W. Burgard. Learning efficient policies for vision-based navigation. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'09)*, St. Louis, USA, 2009.
8. S.J. Julier and J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proceedings of the Int. Symp. on Aerospace/Defense Sensing, Simulation and Controls*, 1997.
9. H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proc. of the Int. Workshop on Augmented Reality (IWAR'99)*, 1999.
10. C. Kwok and D. Fox. Reinforcement learning for sensing strategies. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04)*, Sendai, Japan, 2004.
11. R. Lerner, E. Rivlin, and I. Shimshoni. Landmark selection for task-oriented navigation. *IEEE Transaction on Robotics*, 23(3), 2007.
12. P.S. Maybeck. The Kalman filter: An introduction to concepts. In *Autonomous Robot Vehicle*. Springer Press, 1990.
13. I. Menache, S. Mannor, and N. Shimkin. Basis function adaptation in temporal difference reinforcement learning. *Annals of Operations Research*, 134(1):215–238, 2005.
14. M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simulaneous localization and mapping problem. In *Proc. of the National Conf. on Artificial Intelligence (AAAI'02)*, pages 593 – 598, 2002.
15. C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, USA, 2006.
16. M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The Rprop algorithm. In *Proceedings of the IEEE International Conference on Neural Networks*, 1993.
17. R. Rojas. *Neural Networks: A Systematic Introduction*. Springer Press, 1996.
18. P. Sala, R. Sim, A. Shokoufandeh, and S. Dickinson. Landmark selection for vision-based navigation. *IEEE Transaction on Robotics*, 22(2), 2006.
19. G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, Cambridge, MA, USA, 2006.
20. S.P. Singh, R.S. Sutton, and P. Kaelbling. Reinforcement learning with replacing eligibility traces. In *Machine Learning*, volume 22, pages 123–158, 1996.
21. C. Stachniss, G. Grisetti, W. Burgard, and N. Roy. Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'07)*, San Diego, CA, USA, 2007.
22. 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 and Automation (ICRA'09)*, Kobe, Japan, 2009.
23. R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, USA, 1998.
24. S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Int. Journal of Robotics Research*, 23, 2004.

25. W.T.B. Uther and M. Veloso. Tree based discretization for continuous state space reinforcement learning. In *Proc. of the National Conf. on Artificial Intelligence (AAAI'98)*, pages 769–774, 1998.
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'05)*, 2005.