State Representation Learning in Robotics: Using Prior Knowledge about Physical Interaction

Rico Jonschkowski and Oliver Brock
Robotics and Biology Laboratory, Technische Universität Berlin, Germany

***Abstract*—State representations critically affect the effectiveness of learning in robots. In this paper, we propose a robotics-specific approach to learning such state representations. Robots accomplish tasks by interacting with the physical world. Physics in turn imposes structure on both the changes in the world and on the way robots can effect these changes. Using prior knowledge about interacting with the physical world, robots can learn state representations that are consistent with physics. We identify five robotic priors and explain how they can be used for representation learning. We demonstrate the effectiveness of this approach in a simulated slot car racing task and a simulated navigation task with distracting moving objects. We show that our method extracts task-relevant state representations from high-dimensional observations, even in the presence of task-irrelevant distractions. We also show that the state representations learned by our method greatly improve generalization in reinforcement learning.**

I. Introduction

Creating versatile robots, capable of autonomously solving a wide range of tasks, is a long-term goal in robotics and artificial intelligence. As every one of these tasks might have different sensor requirements, robots must have versatile, task-general sensors, leading to high-dimensional sensory input. These high-dimensional observations, however, present a challenge for perception and learning. This seems unnecessary, as most likely every single task can be mastered by only considering those aspects of the high-dimensional input that are pertinent to it. To build task-general robots, it is therefore necessary to extract from the high-dimensional sensor data only those features pertinent to solving the task at hand.

In robotics, feature engineering is probably the most common approach to this challenge. The mapping from observations to state representation is designed by hand, using human intuition. Feature engineering has enabled systems to successfully learn and solve complex tasks. But the downside of this approach is that we have to define an observation-state-mapping for every robotic task to meet our original goal.

Representation learning methods use machine learning instead of human intuition to extract pertinent information from high-dimensional observations. This approach does not require specific knowledge about the task. Instead, it uses general assumptions about the structure of the problem. However, the price for its generality is the huge amount of data and computation required to extract useful state representations. In robotics, data acquisition is costly and slow. Thus, existing representation learning approaches may be difficult to apply.

But robots do not have to solve the general representation learning problem. Robots only need useful representations for interacting with the real world governed by physics. Physics imposes structure on both the changes in the world and on the way robots can effect these changes. Using prior knowledge about interacting with the world (which we call *robotic priors*), robots can learn representations that are consistent with physics. We believe that prior knowledge is the key to representation learning in robotics.

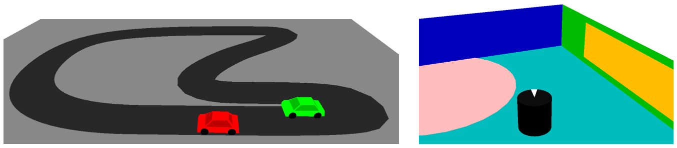

(a) Slot car racing (b) Robot navigation

Fig. 1. Simulated robotic tasks with visual distractors.

In this paper, we identify five robotic priors and explain how they can be used for state representation learning by turning them into a loss function and minimizing this loss function. We evaluate our approach in two simulated robotic tasks based on visual observations: a slot car racing task with two cars and a navigation task with a mobile robot in a room with moving distractors (see Figure 1). In both scenarios, the robot learns a linear mapping from 300-dimensional visual observations to low-dimensional states. We show that the resulting state representation captures the pertinent task dimensions while ignoring irrelevant information. We also show that the observation-state-mapping learned by our method improves generalization and thereby boosts reinforcement learning performance.

II. Related Work

A. Task-Specific, Generic, and Robotic Priors

In Bayesian statistics, the word ”prior” refers to the prior probability distribution that is multiplied by the likelihood and then normalized to compute the posterior. Following others in the field of representation learning [1], we use the word *prior* more broadly in an analogous fashion. In the context of this paper, a prior represents knowledge about a class of learning problems that is available before taking into account data from a specific problem instance. We will now look at different domains, for which priors can be defined.

Many robotic tasks have been successfully solved using reinforcement learning, from ball-in-a-cup to inverted helicopter flight [12]. However, these approaches typically require human engineering, relying on what we call *task-specific priors*, priors that apply only to a specific task. One way of introducing task-specific priors is feature engineering: defining a mapping from observations to task-relevant states by hand.

Work in the area of representation learning strives to remove the need for feature engineering by automatically extracting pertinent features from data. The power of this approach has been empirically demonstrated in tasks such as speech recognition [24], object recognition [13], and natural language processing [4]. All of these examples substantially improve on the best previous methods based on engineered representations. To achieve these results, the representation learning methods use generic priors, big data, and massive computation.

According to Bengio et al. [1], the key to successful representation learning is the incorporation of "many general priors about the world around us." They proposed a list of *generic priors* for artificial intelligence and argue that refining this list and incorporating it into a method for representation learning will bring us closer to artificial intelligence. This is exactly what we are trying to do in the context of robotics. However, we believe that the problem of artificial intelligence is too broad and that therefore generic priors are too weak. We try to find stronger priors about the problem structure by focusing on robotic tasks, which involve interacting with the physical world. We call such priors *robotic priors*.

B. State Representation Learning

State representation learning is an instance of representation learning for interactive problems with the goal to find a mapping from observations to states that allows choosing the right actions. Note that this problem is more difficult than the standard dimensionality reduction problem, addressed by multi-dimensional scaling [14] and other methods [23, 29, 6] because they require knowledge of distances or neighborhood relationships between data samples in state space. The robot, on the other hand, does not know about semantic similarity of sensory input beforehand. In order to know which observations correspond to similar situations with respect to the task, it has to solve the reinforcement learning problem (see Section III), which it cannot solve without a suitable state representation. The question is: What is a good objective for state representation learning? We will now look at different objectives that have been proposed in the literature and relate them to our robotic priors (which we will define in Section IV).

1) Compression of Observations: Lange et al. [15] obtain state representations by compressing observations using deep autoencoders. This approach relies on the prior that there is a simple (low-dimensional) state description and on the prior that this description is a compression of the observations. While we use the same simplicity prior, we believe that it is important to also take time, actions, and rewards into account.

2) Temporal Coherence: Slow feature analysis [30] finds a mapping to states that change as slowly as possible, guided by the prior that many properties in our world change slowly over time. This method has been used to identify a representation of body postures of a humanoid robot [7] as well as for solving reinforcement learning tasks with visual observations [16]. Luciw and Schmidhuber [18] showed that slow feature analysis can approximate proto-value functions [19], which form a compact basis for all value functions. Incorporating the same prior, dimensionality reduction methods have used temporal distance to estimate neighborhood relationships [9].

Slowness or temporal coherence is an important robotic prior that our method also relies on. However, the actions of the robot should also be considered. The following methods and ours take this valuable information into account.

3) Predictive and Predictable Actions: These approaches try to find state representations in which actions correspond to simple, predictable transformations. Action respecting embeddings, proposed by Bowling et al. [3], aim at a state space in which actions are distance-preserving. Sprague's [27] predictive projections try to find a representation such that actions applied to similar states result in similar state changes. Predictive state representations, proposed by Littman et al. [17], define states as success probabilities for a set of tests, where a test is a prediction about future observations conditioned on future actions. Boots et al. [2] showed how predictive state representations can be learned from visual observations. As we will see, these ideas are related to the proportionality prior, the causality prior, and the repeatability prior in this paper.

The problem with these methods, and all other approaches discussed until this point, is that they try to generate task-general state representations. This is problematic when the robot lives in a complex environment and there is no common state representation that works for all tasks. Therefore, we will use the reward to focus on the task-specific aspects of the observations and ignore information irrelevant for the task.

4) Interleaving Representation Learning with Reinforcement Learning: The approaches presented so far learn state representations first to then use them for reinforcement learning. We will now discuss approaches that combine these steps. Piater et al. [21] use decision trees to incrementally discriminate between observations that have inconsistent state-action values according to the reinforcement learning algorithm. This method is comparable to an earlier approach of Singh et al. [26], which minimizes the error in the value function by clustering states. Menache et al. [20] also adapt the state representation during reinforcement learning; they represent the state as a set of basis functions and adapt their parameters in order to improve the value function estimate.

Methods in this category rely on causality of values. They assume that the value is attributable to the state. To compute the value, they must solve the reinforcement learning problem. These steps can be decoupled by factorizing the value function into the reward function and the state transition function. This is used by the following approaches, and also by ours.

5) Simultaneously Learning the Transition Function: In earlier work [11], we proposed to learn the state transition function together with the state representation to maximize state predictability while simultaneously optimizing temporal coherence. A drawback of this approach is that it ignores the reward and therefore cannot distinguish task-relevant from irrelevant information.

6) Simultaneously Learning Transition Function and Reward Function: Some approaches jointly learn an observation-state-mapping, a transition function, and a reward function, differing in their learning objective. Hutter [8] proposes to minimize the combined code length of the mapping, transition function, and reward function. Duell et al. [5] learn these functions to predict future rewards conditioned on future actions. Jetchev et al. [10] maximize state predictability and reward discrimination to learn a symbolic state representation.

These approaches build models of state transitions and rewards to enforce state predictability and reward discrimination. Contrary to this approach, we define our learning objective in terms of distances between state-samples, similar to the idea of multi-dimensional scaling [14]. In this way, we can assure the existence of transition and reward functions for our state representation without having to model them explicitly.

III. State Representation Learning

Reinforcement learning is the problem of learning a policy π to select actions so as to maximize future rewards [28]. The policy π maps states s_t to actions a_t. But as the robot usually cannot directly perceive its current state s_t, it must compute $s_t = \phi(o_t)$ from its observation o_t using an observation-state-mapping ϕ (see Figure 2). Given s_t, the robot performs action $a_t = \pi(s_t)$. This framework describes the interactive loop between the robot and the world. It is therefore well-suited for formalizing many learning problems in robotics.

State representation learning is the problem of learning ϕ, the mapping from observations to states, in order to enable efficient learning of the policy. This is the problem that we address in a robotics-specific way in this paper.

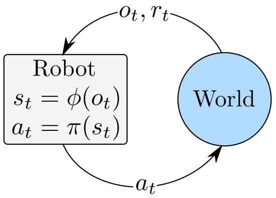


Fig. 2. The robot-world-interaction. At time t, the robot computes the state s_t from its observation o_t using observation-state-mapping ϕ. It chooses action a_t according to policy π with the goal to maximize future rewards $r_{t+1:\infty}$.

Note that the current state s_t could depend on the entire history of observations, actions, and rewards. But in this paper, we assume that the problem is fully observable such that all information required to choose the best action is contained in the last observation. This is a strong limitation, as many real-world problems in robotics are only partially observable. Some—but not all—of the limitations can be alleviated by including sensor inputs from multiple time steps in o_t.

IV. State Representation Learning in Robotics

In this section, we present our approach to state representation learning in robotics. First, we list and explain five robotic priors. Then, we formulate state representation learning as an optimization problem by turning our robotic priors into a loss function. Finally, we turn the theory into a method that tries to minimize this loss function and thereby learns a state representation that reflects our priors.

A. Robotic Priors

The interaction between the robot and the real world is structured by the laws of physics. From this fact, we can derive robotic priors that capture characteristics of robotic tasks.

1) Simplicity Prior: For a given task, only a small number of world properties are relevant. This prior is related to Occam's razor, which is part of the scientific method that aims to form an understanding about our physical world. It favors state representations that exclude irrelevant information, thereby leading to a lower-dimensional reinforcement learning problem and improving generalization.

2) Temporal Coherence Prior: Task-relevant properties of the world change gradually over time. This prior is related to Newton's first law of motion. Physical objects have inertia and change their velocity only gradually as a result of external forces. However, temporal coherence also applies to more abstract properties than physical motion, as most changes in the world occur gradually. The temporal coherence prior favors state representations that obey this principle as the robot transitions between states.

3) Proportionality Prior: The amount of change in task-relevant properties resulting from an action is proportional to the magnitude of the action. This prior results from Newton's second law of motion, $F = m \cdot a$. If an action represents the application of a certain force on an object of a fixed mass, the acceleration evoked by this force is constant. This holds true for motion and physical interactions with objects in the world but also generalizes to more abstract properties. This prior enforces the proportionality principle in the state representation.

4) Causality Prior: The task-relevant properties together with the action determine the reward. This and the next prior resemble Newton's third law of motion or, more generally, causal determinism. If the same action leads to different rewards in two situations, these situations must differ in some task-relevant property and should thus not be represented by the same state. Consequently, this prior favors state representations that include the relevant properties to distinguish these situations.

5) Repeatability Prior: The task-relevant properties and the action together determine the resulting change in these properties. This prior is analogous to the previous one—for states instead of rewards—and also results from Newton's third law of motion. This principle is enforced by favoring state representations in which the consequences of actions are similar if they are repeated in similar situations.

Note that most of these priors are defined in terms of actions and rewards. Thus, they do not apply to passive ”robots” that can only observe but not act. These priors are also not generic artificial intelligence priors applicable to *all* tasks and environments, as artificial environments can be very different from our world, e.g. not obeying Newton's laws of motion.

However, restricting the problem space to the physical world allows us to define useful priors.

But even in the physical world, there are still counterexamples for each prior. In fact, the simulated robotic experiments in this paper include such counterexamples: Proportionality does not hold when the robot is running into a wall and its position remains constant even though it tries to move with a certain velocity. Causality is violated due to sensory aliasing when the robot cannot distinguish two situations with different semantics. Repeatability is contradicted by stochastic actions. Nevertheless, our robotic priors are very useful because they capture the structure of most of the robot's experiences.

B. Formulation as an Optimization Problem

We will now turn the robotic priors into a loss function L such that L is minimized when the state representation is consistent with the priors. We construct loss terms for all robotic priors (except for the simplicity prior, see below) and define L as their sum

$$\begin{aligned} L(D,\hat{\phi}) = L_{\text{temporal coherence}}(D,\hat{\phi}) + L_{\text{proportionality}}(D,\hat{\phi}) \\ + L_{\text{causality}}(D,\hat{\phi}) + L_{\text{repeatability}}(D,\hat{\phi}) \ . \end{aligned}$$

Each of these terms is computed for an observation-state-mapping $\hat{\phi}$ and data of the robot interacting with the world, $D = \{o_t, a_t, r_t\}_{t=1}^{n}$, which consist of sensory observations, actions, and rewards for n consecutive steps. The observation-state-mapping $\hat{\phi}$ is then learned by minimizing $L(D,\hat{\phi})$ (the $\hat{.}$ indicates that ϕ changes during learning).

By linearly combining these loss terms, we assume independence between the robotic priors. They could also be combined non-linearly, but the existence of independent counterexamples for each individual prior supports our assumption. The terms could be weighted differently. However, a linear combination with uniform weights already yields an effective loss function.

We will now describe how the individual robotic priors are defined as loss terms. For better readability, we will write $\hat{s}_t$ instead of $\hat{\phi}(o_t)$ when we refer to the state at time t according to the observation-state-mapping $\hat{\phi}$.

1) Simplicity Loss: The simplicity prior is not formulated as a loss term but implemented by enforcing the state representation to be of fixed, low dimensionality.

2) Temporal Coherence Loss: States must change gradually over time. We denote the state change as $\Delta\hat{s}_t = \hat{s}_{t+1} - \hat{s}_t$. The temporal coherence loss is the expected squared magnitude of the state change,

$$L_{\text{temporal coherence}}(D,\hat{\phi}) = \mathbf{E}\Big[\|\Delta\hat{s}_t\|^2\Big] \ .$$

3) Proportionality Loss: If the robot has performed the same action at times t_1 and t_2, the states must change by the same magnitude $\|\Delta\hat{s}_{t_1}\| = \|\Delta\hat{s}_{t_2}\|$.

The proportionality loss term is

$$L_{\text{proportionality}}(D,\hat{\phi}) = \mathbf{E}\Big[(\|\Delta\hat{s}_{t_2}\| - \|\Delta\hat{s}_{t_1}\|)^2 \ \Big| \ a_{t_1} = a_{t_2}\Big] \ ,$$

the expected squared difference in magnitude of state change after the same action was applied.

4) Causality Loss: Two situations at times t_1 and t_2 must be dissimilar if the robot received different rewards in the following time step, even though it had performed the same action, $a_{t_1} = a_{t_2} \wedge r_{t_1+1} \neq r_{t_2+1}$.

The similarity of two states is 1 if the states are identical and approaches 0 with increasing distance between them. Research from psychology indicates that the exponential of the negative distance is a reasonable similarity function [25], $e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|}$.

We define the causality loss as

$$L_{\text{causality}}(D,\hat{\phi}) = \mathbf{E}\Big[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|} \ \Big| \ a_{t_1} = a_{t_2}, r_{t_1+1} \neq r_{t_2+1}\Big],$$

the expected similarity of the state pairs for which the same action leads to different rewards.

5) Repeatability Loss: States must be changed by the actions in a repeatable way. If the same action was applied at times t_1 and t_2 and these situations are similar (have similar state representations), the state change produced by the actions should be equal, not only in magnitude but also in direction.

We define the repeatability loss term as

$$L_{\text{repeat.}}(D,\hat{\phi}) = \mathbf{E}\Big[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|}\|\Delta\hat{s}_{t_2} - \Delta\hat{s}_{t_1}\|^2 \ \Big| \ a_{t_1} = a_{t_2}\Big] \ ,$$

the expected squared difference in the state change following the same action, weighted by the similarity of the states.

C. Our Method

We will now show how a linear mapping from observations to states can be learned by minimizing the loss function.

1) Computational considerations: We compute the expected values in the loss function by taking the mean over the training samples. For the proportionality loss, the causality loss, and the repeatability loss, this would require comparisons of all $O(n^2)$ pairs of training samples. We approximate these comparisons, for reasons of computational efficiency, by only comparing those samples that are k time steps apart. This way, we can compute the expectations from $O(n)$ pairs of samples. The parameter k does not need careful tuning, it should just be large enough such that states with this temporal distance are roughly uncorrelated. We used $k = 100$ for all experiments.

2) Learning a Linear Mapping from Observations to States: Our method learns a linear observation-state-mapping,

$$\hat{s}_t = \hat{\phi}(o_t) = \hat{W} o_t \ ,$$

where $\hat{W}$ is a weight matrix that is adapted by performing gradient descent on the approximated loss function L. Linear functions form a very limited hypothesis space, but this method can easily be extended to non-linear functions using feature expansion, kernel approaches, or function approximators, such as artificial neural networks.

3) Exploration: Our exploration policy repeats the action from k steps earlier with probability 0.5 and otherwise picks an action at random. This way, the expectation in the loss terms can be estimated using on average at least $\frac{n-k}{2}$ samples. This compensates for the fact that we do not compare *all* pairs of samples but only those k time steps apart.

V. Experiments

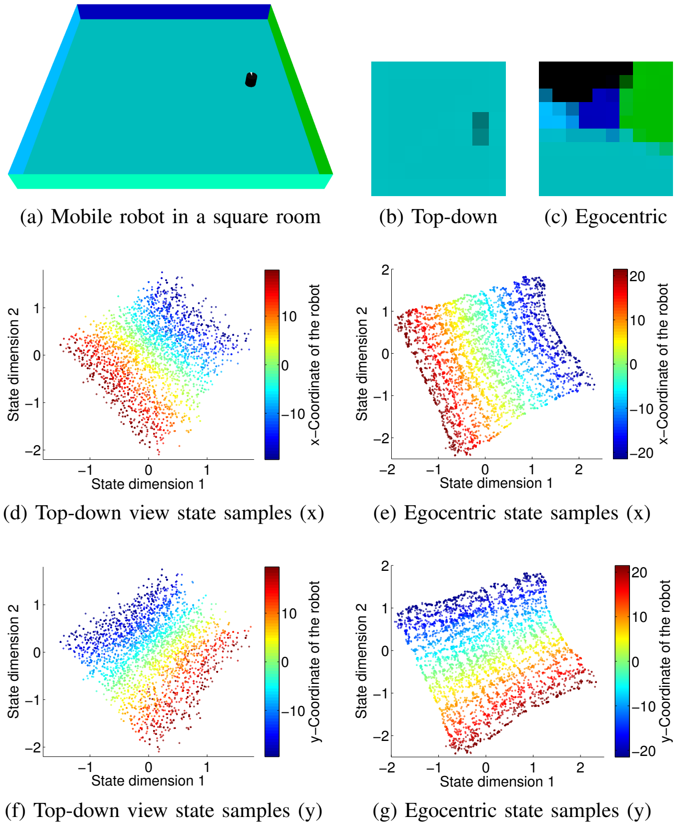
In this section, we evaluate our method in simulated robotic tasks with 300-dimensional visual observations. First, we analyze learned state representations to gain an insight into the capabilities of our approach. We start by comparing learned state representations for a simple navigation task[1] when the robot sees the scene from different perspectives, having either an egocentric view or a top-down view of the scene. The results show that, in both cases, our method learns a mapping to the same pertinent dimensions. Next, we investigate in a slot car racing task[2] how our method can handle task-irrelevant distractors. To the best of our knowledge, this is the first time that this problem is addressed in state representation learning even though it is essential; apart from highly controlled experiments, observations of robots are always subject to task-irrelevant distractions. We will see that our method can separate task-relevant properties of the observation from irrelevant information. After that, we analyze how the state representations for both tasks change if they are given more dimensions than necessary to solve the task. The results show that, in the task without distractors, our method can even identify the minimal state dimensionality.

Finally, we measure how useful the learned state representations really are for subsequent reinforcement learning, as this is the main motivation for state representation learning. We extend the navigation task and also introduce distractors to make it more challenging. We compare a standard reinforcement learning method on different state representations. The experiment shows that our method can substantially improve the performance of reinforcement learning compared to different baselines. In a last experiment, we explain these results by showing how our approach improves generalization.

A. Invariance to Perspective

To investigate whether our method is invariant to perspective, we test it in two versions of a simple navigation task with different visual observations, viewing the scene from the top and viewing it from the robot's perspective. In both versions, the robot learns a state representation that reflects its location which is exactly the information required to solve the task.

1) The Simple Navigation Task: In the simple navigation task (see Figure 3a), the robot is located in a square-shaped room of size 45×45 units with 4-units-high walls of different colors. The robot has a height and diameter of 2 units. The orientation of the robot is fixed but it can control its up-down and left-right velocity choosing from $[-6, -3, 0, 3, 6]$ units per time step. The robot thus has 25 discrete actions. These actions are subject to Gaussian noise with 0 mean and standard deviation of 10% of the commanded velocity. The task of the robot is to move to the top right corner without bumping into walls. If the distance to this corner is less than 15 units, the robot gets a reward $+10$ unless it is running into a wall, in which case it gets a negative reward of -1. The observation of the robot is a 10×10-pixel RGB image. In the top-down view version of this task, the robot's observation is an image of the entire room. In this image, the robot is a dark spot against the background (see Figure 3b). In the egocentric version, the robot perceives its environment through a camera with a wide angle lens (field of view 300°). The example observation (see Figure 3c) shows the dark blue wall in the middle and the green and the light blue wall on either side of the image.



(a) Mobile robot in a square room (b) Top-down (c) Egocentric

(d) Top-down view state samples (x) (e) Egocentric state samples (x)

(f) Top-down view state samples (y) (g) Egocentric state samples (y)

Fig. 3. Results for two versions of the simple navigation task with fixed orientation. The observation of the robot is either a top-down view (b) or an egocentric view (c) of the scene (a). (d–g) show the representation of 5000 training samples in state space. Each dot results from applying the learned observation-state-mapping ϕ to an observation. (d,f) correspond to the top-down view version, (e,g) correspond to the egocentric version of the task. The color relates the state samples to the ground truth location of the robot.

2) Experimental Setup: We performed the following experiment for both versions of the task. The robot explored its environment performing 5000 random actions and learned a mapping from the 300-dimensional observation space to a two-dimensional state representation based on this experience.

3) Results: To compare the learned state representations, we plotted the state estimates for these 5000 time steps for the top-down view (see Figure 3d) and the egocentric view (see Figure 3e). In both cases, the samples roughly form a square in state space, suggesting that the state is an estimate of the location of the robot in the square room. We can show that this is in fact what is learned by coloring each state sample according to the ground truth x-coordinate of the robot (see Figures 3d and 3e) and to the ground truth y-coordinate of the robot (see Figures 3f and 3g). The results are the same

[1] The navigation task is based on similar experiments in the literature [2, 27].

[2] The slot car racing task is inspired by an experiment of Lange et al. [15].

for both state representations: there are two orthogonal axes in the state representations that correspond to the coordinates of the robot. Of course, these axes in state space do not have to align between experiments; they can be rotated or flipped.

4) Discussion: In the two versions of the task, the sensory observations of the robot were very different. Nevertheless, it learned a mapping from these observations to the task-relevant dimensions—the location of the robot. Note that the mapping from observations to states must be very different to result in this identical state representation.

B. Ignoring Distractors

In this experiment, we test whether our method distinguishes task-relevant properties of the observations from irrelevant information. We investigate this in a slot car racing task with two cars. While the robot observes two slot cars, it can only control one of them. The other car does not play a role in this task apart from potentially distracting the robot. The robot does not know beforehand which car is relevant for the task.

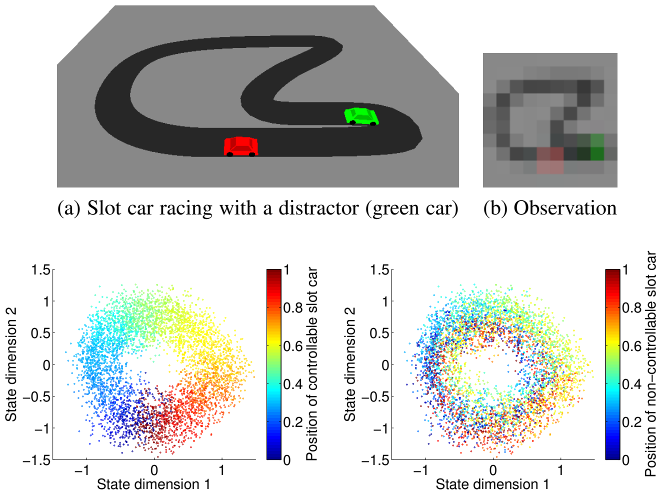
1) The Slot Car Racing Task: An example scene from this task is shown in Figure 4a. The robot can control the velocity of the red car, choosing from $[0.01, 0.02, \ldots, 0.1]$ units per time step. The velocity is subject to zero mean Gaussian noise with standard deviation of 10% of the commanded velocity. The robot's reward is equal to the commanded velocity—unless the car goes too fast in a sharp turn and is thrown off the track. In this case, the robot gets a reward of -10. The robot cannot control the green slot car. The velocity of this car is chosen randomly from the same range as for the red car. The green slot car does not influence the reward of the robot or the movement of the red car. The robot observes the scene from the top through a 10×10-pixel RGB image (see Figure 4b).

2) Experimental Setup: The robot explored randomly for 5000 time steps and then learned a mapping from its 300-dimensional observation to a two-dimensional state.

3) Results: To understand this state representation, we have plotted the states of the 5000 exploration steps with one dot per state sample (see Figure 4c). The states form a circle which corresponds to the topology of the track. We have colored the state samples according to the ground truth position of the red slot car (see Figure 4c) and the green slot car (see Figure 4d). The figures show that the position along this circle in state space corresponds to the position of the controllable slot car on the track. One round of the red slot car corresponds to a circular trajectory in state space. Our method was able to distinguish task-relevant from irrelevant information in the observations and, at the same time, found a compressed representation of these pertinent properties.

C. Mapping to a Higher-Dimensional State Space

In the previous experiments, we gave the robot an appropriate number of dimensions for the state representation. In this section, we investigate what happens in the same examples, when the robot tries to learn state representations with more dimensions than necessary. We repeated the experiments for the simple navigation task with egocentric observations and for the slot car task. But instead of learning a two-dimensional state representation, we used a five-dimensional state space. After exploration for 5000 time steps and state representation learning, we took the 5000×5-matrix M containing the estimated states for these experiences and performed a principal component analysis of this matrix.

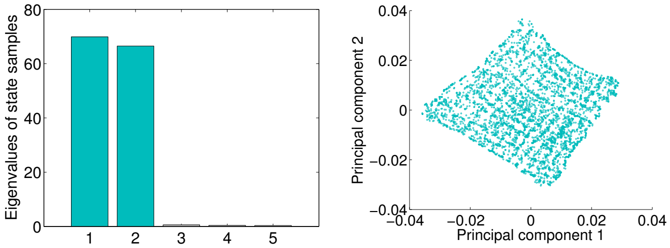


(a) Slot car racing with a distractor (green car) (b) Observation

(c) State samples (red car) (d) State samples (green car)

Fig. 4. Results for the slot car racing task (a) with visual observations (b). The color relates state samples to the relevant car (c) and the distractor (d).

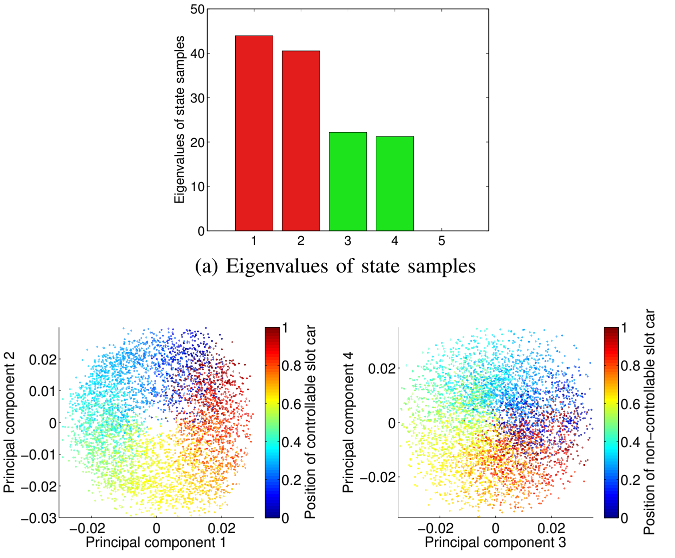
1) Identifying the Dimensionality of the Task: For the navigation task, we find that all but the first two eigenvalues of M are close to zero (see Figure 5a). The rank of the matrix is effectively two. This means that all state samples lie on a plane in the five-dimensional state space. We can visualize this plane by projecting the state samples on their first two principal components (see Figure 5b). The state samples again form a square in this space just as in the two-dimensional experiment. Thus, even with a five-dimensional state space, the robot learns that the task is two-dimensional and captures only those properties of its observation.



(a) Eigenvalues of state samples (b) Projected state samples

Fig. 5. Results for the navigation task with a five-dimensional state space.

2) Finding Alternative Explanations for the Reward: In the slot car task, the state sample matrix M has rank four. There are two larger eigenvalues and two smaller eigenvalues (see Figure 6a). If we project the state samples on their first two principal components, we can see that the dimensions



(a) Eigenvalues of state samples

(b) Projected state samples (red car) (c) Projected state samples (green car)

Fig. 6. Results for the slot car task with a five-dimensional state space.

with the larger eigenvalues correspond to the position of the controllable red slot car on the race track (see Figure 6b). The third and fourth principal component correspond to the position of the non-controllable green slot car (see Figure 6c).

3) Discussion: If the green car is irrelevant for the task, why is it represented in the state? The robot maximizes state dissimilarity between situations where it received different rewards even though it performed the same action. If the robot chooses the same velocity but the slot car is thrown off one time while it stays on track another time, it tries to make the states of these two situations dissimilar. The most powerful discrimination between these situations is the position of the red slot car. But sometimes small differences in position or the stochasticity of the actions can make the difference between the two outcomes. The robot thus finds alternative explanations like the position of the green slot car. The eigenvalues show that this property has a lower impact on the state than the position of the controllable red slot car. Our method includes these alternative explanations if there are enough dimensions in the state space. When the state space is limited, the method focuses on pertinent dimensions.

D. Improved Performance in Reinforcement Learning

The preceding experiments have demonstrated promising properties of our method. But in the end, the utility of state representations can only be measured by how they benefit subsequent learning. In this experiment, we will see that our method can substantially improve reinforcement learning performance and that it needs very few data to do so.

1) The Extended Navigation Task: To construct a challenging task for this experiment, we extended the navigation task by allowing variable orientation of the robot and adding visual distractors. The robot can turn and move forwards or backwards choosing its rotational velocity from $[-30, -15, 0, 15, 30]$ degrees per time step and its translational velocity from $[-6, -3, 0, 3, 6]$ units per time step for a total of 25 actions. All actions are subject to Gaussian noise with 0 mean and 10% standard deviation. The distractors are three circles on the floor and four rectangles on the walls that move randomly (see Figure 7). The distractors are observed by the robot but do not influence its movement or reward. They are irrelevant for the task and should thus not be included in the state representation. The objective of the task has not changed. The robot must move to within 15 units of the top right corner, where it gets a reward of 10 unless it is running into a wall, in which case it gets a reward of -1.

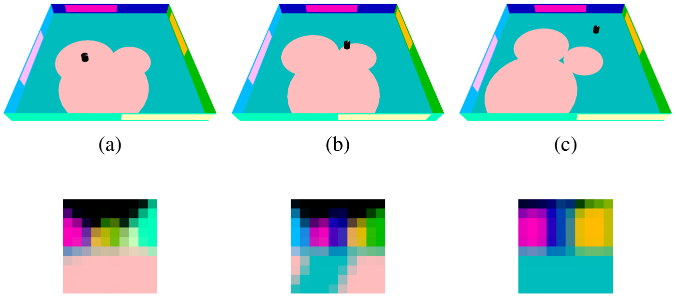


Fig. 7. Extended navigation task. (a-c) show the robot moving to the upper right corner while the distractors move randomly. (d-f) show the respective observations (note how they are influenced by the distractors).

2) Experimental Setup: The robot explored randomly as in previous experiments but was interrupted every 500 time steps. From its accumulated experience, it learned an observation-state-mapping and a policy. Then, the robot was tested for 20 episodes of 50 steps to compute the average sum of rewards. We repeated this learning-evaluation cycle ten times.

We conducted this experiment multiple times using the same reinforcement learning method with different state representations: the five-dimensional state representation learned with our method, the five slowest features of the observations (computed using linear slow feature analysis [30]), the first five principal components of the observations, and the raw 300-dimensional observation. To get an upper bound on the reinforcement learning performance, we also compared against a simpler version of this task without distractors in which the robot has access to its ground truth pose. In this case it uses its position and the cosine and sine of its orientation as state, which we consider an optimal representation for this task.

3) Reinforcement Learning Method: As a reinforcement learning method, we used neural fitted Q-iteration [22] with the default parameters on a neural network with two hidden layers, each containing five sigmoid neurons.

4) Results—Improved Generalization: We want to start analyzing the results in Figure 8, by comparing our method (green) against using the raw observations as states directly (orange). These results show very clearly that the robot needs much less training when reinforcement learning is performed on the states learned by our method. Where does this difference come from? Our method basically acts as a regularization on the learning problem. This leads to faster generalization.

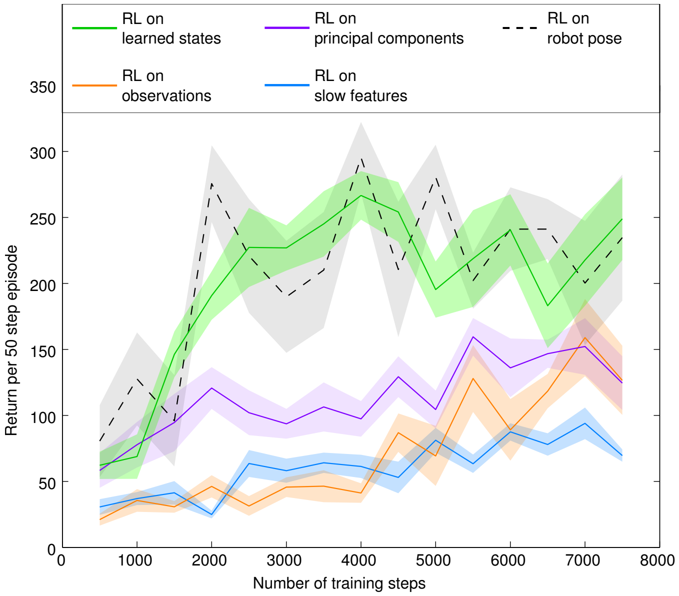


Fig. 8. Reinforcement learning performance for different state representations. Lines show means, surfaces display their standard errors.

We examine this in more detail with another experiment—this time in a supervised learning task, which allows us to visualize generalization by comparing training error and test error. In this experiment, we compare how well the location of the robot can be estimated from the learned state representation and from the raw observation. The location estimator is learned by linear regression from a set of training samples which contain the ground truth coordinates and either the 300-dimensional observations or five-dimensional states (learned from 5000 exploration steps).

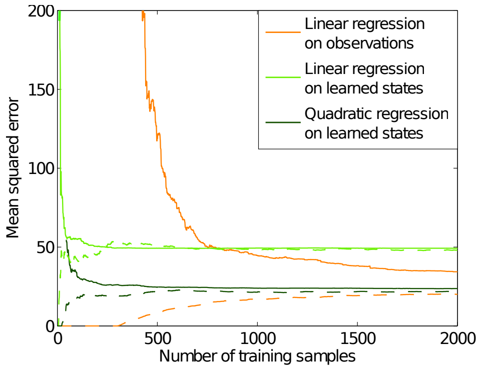


Fig. 9. Learning curves for location estimation. Solid lines display the test error, dashed lines display the training error.

The learning curves (see Figure 9) show that linear regression on the learned states (light green) generalizes very quickly—having the same error on the training set (dashed line) as on the test set (solid line). Generalizing based on raw observations (orange) takes much more training. But with enough training, its test performance surpasses the performance based on the learned state representation, which shows that a linear function in the state space is probably too restricted. Fitting a more complex quadratic function in the state space also needs very few training to generalize and results in superior performance (dark green).

The key to these results is the compactness of the representation. A linear function in the state space has six parameters (one for each state dimension plus one for the bias), a quadratic function has 21 parameters. A linear function in observation space has 301 parameters. Too many parameters lead to overfitting: the training error is low, but the test error is high. The same principle applies to reinforcement learning.

5) Results—Pertinent State Representation: This generalization argument, however, does not explain the performance differences between our approach and the baselines: principal component analysis (purple) and slow feature analysis (blue, see Figure 8). All of these representations have the same dimensionality. The other methods probably did not perform well because they failed to distinguish relevant from irrelevant dimensions of the task as they are purely based on observations and do not take actions and rewards into account.

6) Results—Little Training Required: Finally, we compare the results of our approach to the upper bound of the reinforcement learning method—using the ground truth pose of the robot as state (dashed line, see Figure 8). Keep in mind how much easier this task is compared to coping with 300-dimensional visual observations influenced by distractors. Still, the results show that our method is able to learn a state representation that is as useful for reinforcement learning as the true pose of the robot. Even with very little training, the results are comparable showing that our method actually needs less data than the reinforcement learning method.

VI. Discussion

We have presented an approach to state representation learning in robotics based on prior knowledge about interacting with the physical world. The first key idea to this approach is to focus on state representation learning in the physical world instead of trying to solve the general problem of state representation learning in arbitrary artificial environments. Reducing the problem domain in this way allows us to use robotics-specific prior knowledge.

The second key idea is to use this knowledge to evaluate representations by how consistent they are with our priors about the world. We proposed five robotic priors—simplicity, temporal coherence, proportionality, causality, and repeatability—and showed how they can be turned into an objective for state representation learning.

We would like to advocate formulating additional priors about inherent problem structure in robotics in future research. Hopefully, this will lead to a discussion on the ”right” robotic priors. Additionally, we think that the field should strive to find new ways to use robotic priors in machine learning.

VII. Acknowledgments

We gratefully acknowledge financial support by the Alexander von Humboldt foundation through an Alexander von Humboldt professorship (funded by the German Federal Ministry of Education and Research).

References

[1] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[2] Byron Boots, Sajid M. Siddiqi, and Geoffrey J. Gordon. Closing the learning-planning loop with predictive state representations. *International Journal of Robotics Research*, 30(7):954–966, 2011.

[3] Michael Bowling, Ali Ghodsi, and Dana Wilkinson. Action respecting embedding. In *22nd International Conference on Machine Learning (ICML)*, pages 65–72, 2005.

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[5] Siegmund Duell, Steffen Udluft, and Volkmar Sterzing. Solving partially observable reinforcement learning problems with recurrent neural networks. In *Neural Networks: Tricks of the Trade*, volume 7700 of *Lecture Notes in Computer Science*, pages 709–733. Springer Berlin Heidelberg, 2012.

[6] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1735–1742, 2006.

[7] Sebastian Höfer, Manfred Hild, and Matthias Kubisch. Using slow feature analysis to extract behavioural manifolds related to humanoid robot postures. In *10th International Conference on Epigenetic Robotics*, pages 43–50, 2010.

[8] Marcus Hutter. Feature reinforcement learning: Part I: Unstructured MDPs. *Journal of Artificial General Intelligence*, 1:3–24, 2009.

[9] Odest Chadwicke Jenkins and Maja J. Matarić. A spatio-temporal extension to isomap nonlinear dimension reduction. In *21st International Conference on Machine Learning (ICML)*, page 56, 2004.

[10] Nikolay Jetchev, Tobias Lang, and Marc Toussaint. Learning grounded relational symbols from continuous data for abstract reasoning. In *Autonomous Learning Workshop at the IEEE International Conference on Robotics and Automation*, 2013.

[11] Rico Jonschkowski and Oliver Brock. Learning task-specific state representations by maximizing slowness and predictability. In *6th International Workshop on Evolutionary and Reinforcement Learning for Autonomous Robot Systems (ERLARS)*, 2013.

[12] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1106–1114, 2012.

[14] Joseph B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[15] Sascha Lange, Martin Riedmiller, and Arne Voigtländer. Autonomous reinforcement learning on raw visual input data in a real world application. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2012.

[16] Robert Legenstein, Niko Wilbert, and Laurenz Wiskott. Reinforcement learning on slow features of high-dimensional input streams. *PLoS Computational Biology*, 6(8):e1000894, 2010.

[17] Michael L. Littman, Richard S. Sutton, and Satinder Singh. Predictive representations of state. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1555–1561, 2002.

[18] Matthew Luciw and Juergen Schmidhuber. Low complexity proto-value function learning from sensory observations with incremental slow feature analysis. In *22nd International Conference on Artificial Neural Networks and Machine Learning (ICANN)*, pages 279–287, 2012.

[19] Sridhar Mahadevan and Mauro Maggioni. Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 8(10):2169–2231, 2007.

[20] Ishai Menache, Shie Mannor, and Nahum Shimkin. Basis function adaptation in temporal difference reinforcement learning. *Annals of Operations Research*, 134:215–238, 2005.

[21] Justus Piater, Sébastien Jodogne, Renaud Detry, Dirk Kraft, Norbert Krüger, Oliver Kroemer, and Jan Peters. Learning visual representations for perception-action systems. *International Journal of Robotics Research*, 30(3): 294–307, 2011.

[22] Martin Riedmiller. Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method. In *16th European Conference on Machine Learning (ECML)*, pages 317–328, 2005.

[23] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[24] Frank Seide, Gang Li, and Dong Yu. Conversational speech transcription using context-dependent deep neural networks. In *Interspeech*, pages 437–440, 2011.

[25] Roger N. Shepard. Toward a universal law of generalization for psychological science. *Science*, 237(4820): 1317–1323, 1987.

[26] Satinder P. Singh, Tommi Jaakkola, and Michael I. Jordan. Reinforcement learning with soft state aggregation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 361–368, 1995.

[27] Nathan Sprague. Predictive projections. In *21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1223–1229, 2009.

[28] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[29] Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[30] Laurenz Wiskott and Terrence J. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.