Learning Predictive State Representations for Planning

Johannes A. Stork Carl Henrik Ek Danica Kragic

***Abstract*— Predictive State Representations (PSRs) allow modeling of dynamical systems directly in observables and without relying on latent variable representations. A problem that arises from learning PSRs is that it is often hard to attribute semantic meaning to the learned representation. This makes generalization and planning in PSRs challenging. In this paper, we extend PSRs and introduce the notion of PSRs that include prior information (P-PSRs) to learn representations which are suitable for planning and interpretation. By learning a low-dimensional embedding of test features we map belief points of similar semantic to the same region of a subspace. This facilitates better generalization for planning and semantical interpretation of the learned representation. In specific, we show how to overcome the training sample bias and introduce feature selection such that the resulting representation emphasizes observables related to the planning task. We show that our P-PSRs result in qualitatively meaningful representations and present quantitative results that indicate improved suitability for planning.**

I. INTRODUCTION

Intelligence can be seen as the capability to act in an appropriate manner when the knowledge provided from the environment is uncertain. Humans achieve this by—among other things—reducing uncertainty through incorporation of previous experience and knowledge. Likewise, robotics as a field strives to create intelligent artificial agents that are capable of acting when the state of the world is uncertain. To accomplish this goal we need to investigate how to best utilize prior knowledge.

It has been argued that prior knowledge should consist of architectural constraints and fundamental truths such as physical laws. The remaining concepts—such as the world model or sensory-motor loops—must be *learned* by the agent with the help of priors [12]. A list of generic priors for representation learning in AI has been proposed by [1], while arguing that the success of machine learning algorithms generally depends on data representation. However, these generic priors have been qualified as too weak and stronger and more specific priors have been proposed for representation learning for robots [11], such as *causality* and *temporal coherence*.

However, in order to incorporate such priors we need to devise a representation of the environment, the current state of the robot, and the system dynamics. In many scenarios this task is very challenging. Predictive State Representations (PSRs) [16] describe the state of the agent directly in terms of observable quantities thereby implicitly connecting the environment and the agent's state. The model tries to learn a representation of the environment such that the future state can be predicted successfully. Directly representing the system in terms of observables allows the agent to learn a representation by exploring the environment and observing the change induced by its actions. Having conceived a representation, plans can be formulated and executed directly in terms of observable quantities. This *closing of the loop* [3] was initially proposed as a motivation for using PSRs, because the state space could first be learned by a simple exploration phase and later be used to formulate plans. In this paper we argue that planning and representation learning should not be decoupled in this manner. Dependent on the plan to execute, different observable quantities are of different importance and the representation should reflect this to support planning. For example: The iconic London Tube map (see Fig. 1) is an excellent tool for planning travel when your objective is to get to a specific location, however it is not a good tool for estimating travel time as stations are depicted as equidistant on the map which is far from true geographically. Therefore, a good representation for planning is one that captures the relevant information for the specific planning task.

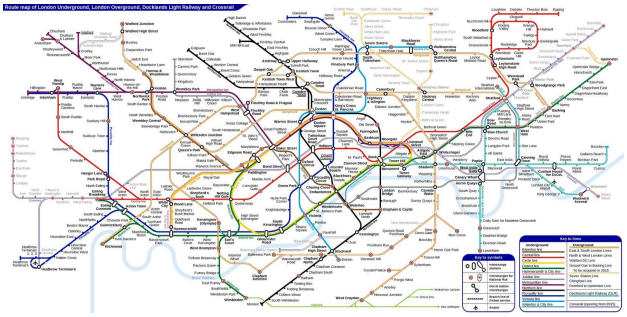


Fig. 1. *The London Tube map is a representation specifically designed to for planning of underground travel. Rather than depicting the true geographical locations, the map provides a different semantic; stations are evenly spaced; transit stations are emphasized; different lines are separated; simple primitives connect stations. A traveler should be able to easily identify current and target station which is aided by the even spacing, color coding makes it easy to see if a transit is necessary and places where this is possible are emphasized. Once the train has embarked no action is possible till the next stop, therefore actual geographical positions of tracks between stations are irrelevant.*

In this paper we extend PSRs to exploit prior information about the desired planning task, called P-PSRs. Our approach is analogous to metric learning [30] where we pose geometric preferences on the learning problem. In summary, our work brings priors into the field of subspace identification-based PSR learning and for the first time allows for informed constraints on the features that form the PSR state.

The authors are with the Computer Vision and Active Perception Lab, Centre for Autonomous Systems, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden, {jastork,chek,dani}@kth.se. This work was supported by FLEXBOT (FP7-ERC-279933), the Swedish Research Council and the Swedish Foundation for Strategic Research.

II. RELATED WORK

For robots and many other AI agents, planning and taking actions happen in continuous environments under uncertainty which makes the concepts of belief and state parameterization relevant [13, 26]. Recent reinforcement learning methods address continuous domains by approximating a so-called $\boldsymbol{Q}$-function from a sparse set of experiences using supervised learning. They are therefore concerned with exploiting (global) generalization—by assigning similar values to related areas—while at the same time avoiding opposing effects. This *fitted $\boldsymbol{Q}$-iteration* can use randomized trees [6] or multi-layer perceptrons [20] for regression. Impaired generalization can result in long learning times or in final failure [20]. A representation or model which facilitates regression of $\boldsymbol{Q}$-values is therefor desirable. In this work, we learn PSR models with priors such that semantically similar states are located in the same region which accommodates regression. $\boldsymbol{Q}$-fitted learning has already been applied to discrete PSRs [18, 8] while only point-based value iteration [10] has been used for continuous domain PSRs [3]. Here, we use reinforcement learning that builds directly on neighbor look-up in observed experiences as in our previous work[25].

Modeling the world as a *Partially Observable Markov Decision Process* (POMDP) is common in robotics and it allows reasoning about uncertainty in action effects and state observability. However, to *a-priori* define a latent variable representation is very challenging. It requires that the complete system model is known, and such expert knowledge can be sparse or biased [8]. In many scenarios it is therefore beneficial if a representation can be inferred from the data automatically [3, 11, 8, 17].

Learning latent state models from *exploration data* had only limited success, because of the *curse of dimensionality* of observations and the *curse of history* of required training sequences [3]. PSR [16, 23] is a different approach to model high-dimensional dynamical systems that represents world states as predictions about future events which are directly observable quantities (as compared to POMDP latent states). Therefore, they can be constructed without the expectation maximization-type heuristics for POMDPs, which are prone to local minima.

Learning a PSR consists of two challenges: Discovery of a set of future events to form a sufficient state, and learning state update parameters. Most frequently, the discovery problem is addressed as subspace identification called transformed PSR (TPSR) which exploits linearity of the model [21]. It allows parameter learning by regression from empirical estimates of observable quantities. Other approaches consider embedding of nonparametric distributions [4] or learn exponential family distributions of tests [27]. In this work, we consider the subspace identification approach and learn the embedding mapping.

The subspace can be created by engineering a set of sufficient features, or by selecting from a larger amount of sampled or randomly generated features. In the latter case, the subspace can be found by a lower-dimensional embedding with spectral methods [21, 3, 2], approximately preserving feature distances [9, 8], or by applying regularized noise-removal to identify the true dimensionality of the underlying system [7]. None of these methods take the subsequent planning task based on the learned model in due consideration. In contrast, we optimize a low-dimensional feature embedding to satisfy certain priors on the subspace that facilitate planning.

The authors of [11] argue that embodied agents like robots accomplish tasks by interactions with the environment and that physics imposes structure on world changes and actions effects. A robotic state representation should therefore be consistent with these constraints. Similar to our work, they learn a linear embedding and apply priors that describe the evolution of the world in time to learn pertinent features. Different to us, they operate in an observable domain and avoid explicit knowledge about the semantic relationship between samples in the form of informed labels. We use label distances to encourage consistency with relevant partially observable concepts. Other implementations of priors employ deep auto encoders [15], slow feature analysis [28], or constraints on effects [5] or consistency [24] of actions. Action semantics are not involved in our priors but our methodology allows for such priors. Similar to the labels in our approach, [22] use weak geometric information to learn a probabilistic topological model for robot navigation. Odometric information is integrated as a transition observation that has to satisfy geometrical constraints. This kind of technique has not been explored for PSRs yet.

III. BACKGROUND

In this section, we overview the transformed PSR formalism and subspace learning method which have been used in several other works before [21, 3, 2, 9, 8, 7]. A more detailed description of the TPSR theory can be found in [3, 8] from where we adopt notation and designations.

A PSR represents the state of a partially observable system as probability distributions over a set of future events conditioned on a history of previous events [16, 23]. Future events and histories are formalized as sequences of actions and observations. An observed sequence of actions a_i and corresponding observations o_i is a history $h = (a_0, o_0, a_1, o_1, \ldots, a_t, o_t)$. A sequence of observations conditioned on interventions (i.e. performing an action) of the agent is referred to as a test $\tau = (o_{t+1}, o_{t+2}, \ldots, o_{t+n} || a_{t+1}, a_{t+2}, \ldots, a_{t+n})$. In both cases actions $a_i \in \mathcal{A}$, observations $o_i \in \mathcal{O}$, and subscripts refer to their respective order in time. The symbol $||$ denotes the agent's intervention or its plan [19] and we write the probability for the future event τ as $P(\tau|h) = P(o_{t+1}, \ldots, o_{t+n}|h||a_{t+1}, \ldots, a_{t+n}) = P(\tau^{\mathcal{O}}|h||\tau^{\mathcal{A}})$.

The underlying assumption for *linear* PSRs is that statistics over a (minimal) *core set* of tests $\mathcal{Q}$, allow to reconstruct the probabilities for all possible tests using a set of *linear functions*. These functions map the probabilities of the core set to probabilities of all tests. Denoting the probabilities of the core set by the vector $\mathbf{m}_t$ and by collecting the linear

functions as weight vectors in matrices $\mathbf{M}_{ao}$, the linear PSR state can be maintained by a vector-valued equation,

$$\mathbf{m}_{t+1} = \frac{\mathbf{M}_{ao}\mathbf{m}_t}{\mathbf{m}_\infty^\intercal \mathbf{M}_{ao}\mathbf{m}_t}, \tag{1}$$

where $\mathbf{m}_\infty$ is a normalizing factor to ensure that $\mathbf{m}_{t+1}$ can be interpreted as a vector of probabilities.

For complex problems it is often not feasible to delineate a core set that provides sufficient statistics for the whole problem domain. A method referred to as transformed PSRs [21] takes another approach. Rather than finding a subset of tests it aims to identify a linear subspace of a set of tests $\mathcal{T}$ that provides sufficient statistics. The TPSR is formalized by the transformed parameters of the linear PSR,

$$\mathbf{Z}\mathbf{m}_{t+1} = \frac{\mathbf{Z}\mathbf{M}_{ao}\mathbf{Z}^{-1}\mathbf{Z}\mathbf{m}_t}{((\mathbf{Z}^{-1})^\intercal \mathbf{m}_\infty)^\intercal \mathbf{Z}\mathbf{M}_{ao}\mathbf{Z}^{-1}\mathbf{Z}\mathbf{m}_t} = \frac{\mathbf{B}_{ao}\boldsymbol{\beta}_t}{\boldsymbol{\beta}_\infty^\intercal \mathbf{B}_{ao}\boldsymbol{\beta}_t}, \tag{2}$$

where a matrix $\mathbf{Z}$ extends Eq. (1) to the update of the transformed state for a new action-observation pair ao. The projection matrix $\mathbf{Z}$ consists of two parts and factorizes as $\mathbf{Z} = \mathbf{J}\mathbf{R}$ where $\mathbf{R}$ maps core set test probabilities to probabilities of tests in $\mathcal{T}$ and $\mathbf{J}$ defines the change of basis to the subspace of sufficient statistics.

A. TPSRs with Features

The histories and test above are discrete entities. In order to apply PSRs for continous domains an extension that works with features of tests $\phi^{\mathcal{T}}$ and histories $\phi^{\mathcal{H}}$ has been introduced [3]. To include features, *observable matrices* are defined in terms of feature expectations: The marginal expected history features $\boldsymbol{P}_{\mathcal{H}}$, the expected product of tests and history features $\boldsymbol{P}_{\mathcal{T},\mathcal{H}}$, and the expected feature product for tests and history which are appended by the action and observation pair ao, $\boldsymbol{P}_{\mathcal{T},ao,\mathcal{H}}$:

$$\begin{aligned}
(\boldsymbol{P}_{\mathcal{H}})_i &= \mathbb{E}(\phi_i^{\mathcal{H}}) \\
(\boldsymbol{P}_{\mathcal{T},\mathcal{H}})_{i,j} &= \mathbb{E}(\phi_{i,t}^{\mathcal{T}}\phi_{j,t}^{\mathcal{H}} || (\phi_i^{\mathcal{T}})^{\mathcal{A}}) \\
(\boldsymbol{P}_{\mathcal{T},ao,\mathcal{H}})_{i,j} &= \mathbb{E}(\phi_{i,t+1}^{\mathcal{T}}\mathbb{I}(o_t{=}o)\phi_{j,t}^{\mathcal{H}} || a_t{=}a, (\phi_{i,t+1}^{\mathcal{T}})^{\mathcal{A}}).
\end{aligned} \tag{3}$$

When using features, the projection matrix $\mathbf{Z}$ will include an additional term $\mathbf{Z} = \mathbf{J}\Phi^{\mathcal{T}}\mathbf{R}$ where the matrix $\Phi^{\mathcal{T}}$ contains the weights of the tests in $\mathcal{T}$ and maps from test space to feature space. Consequently, by observing Eq. (2) the state $\mathbf{b}_t = (\mathbf{J}\Phi^{\mathcal{T}}\mathbf{R})\mathbf{m}_t$ of a feature-based TPSR is the expectation of test features that are linearly combined by $\mathbf{J}$.

B. Learning TPSRs

TPSR learning builds *empirical estimates* of the *observable matrices* (denoted here by $\hat{\cdot}$) for a set of test and history features. Learning the TPSR parameters can be formulated as a regression problem in the form of a Moore-Penrose pseudo inverse. In this case, the linear PSR parameters are learned in a statistically consistent way [3]:

$$\hat{\boldsymbol{\beta}}_0 = \mathbf{J}\hat{\boldsymbol{P}}_{\mathcal{T},\mathcal{H}}\mathbf{1}_k \tag{4}$$

$$\hat{\boldsymbol{\beta}}_\infty^\intercal = \hat{\boldsymbol{P}}_{\mathcal{H}}^\intercal(\mathbf{J}\hat{\boldsymbol{P}}_{\mathcal{T},\mathcal{H}})^+ \tag{5}$$

$$\hat{\mathbf{B}}_{ao} = \mathbf{J}\hat{\boldsymbol{P}}_{\mathcal{T},ao,\mathcal{H}}(\mathbf{J}\hat{\boldsymbol{P}}_{\mathcal{T},\mathcal{H}})^+ \quad . \tag{6}$$

As a consequence of this formulation, the projection matrix $\mathbf{J}$ has to be chosen such that $\mathbf{Z} = \mathbf{J}\Phi^{\mathcal{T}}\mathbf{R}$ is invertible and the pseudo inverse $(\mathbf{J}\boldsymbol{P}_{\mathcal{T},\mathcal{H}})^+$ has full row rank. An evident choice is $\mathbf{J} = \mathbf{U}^\intercal$ from the singular value decomposition (SVD) of $\mathbf{P}_{\mathcal{T},\mathcal{H}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\intercal$ because it gives $(\mathbf{J}\boldsymbol{P}_{\mathcal{T},\mathcal{H}})^+ = \mathbf{V}\boldsymbol{\Sigma}^{-1}$. In the remainder of this paper, we refer to this as *spectral method* since the truncated SVD provides us with a spectral embedding.

IV. METHODOLOGY

Our goal is to learn a TPSR that is optimized for semantical interpretation of geometry, exploitation for geometry-based heuristics, and generalization for belief space planning. This is achieved when distances in the learned representation mirror the semantics in the true system or beliefs over them. To this end, we learn a test feature embedding subject to priors. Below, we formulate our learning problem as an optimization problem, describe the optimization parameters, the loss-functions, and the training data.

A. Projecting Features of Tests

As explained in Sec. III-A, a feature-based TPSR represents state as linear combinations of test feature expectations. Usually, the number of test features $n_{\mathcal{T}}$ is large (e.g. several hundred) and more than sufficient for the actual low-dimensional system dynamics. Therefore, the matrix $\mathbf{J}$ projects to a low-dimensional embedding space $\mathcal{S}$.

$$\mathbf{J}\colon \mathbb{R}^{n_{\mathcal{T}}} \to \mathbb{R}^d, \qquad \mathbf{J}\colon \phi^{\mathcal{T}} \mapsto \mathbf{s} \tag{7}$$

Consequently, $\mathbf{J}$ controls the *dimensionality* and *structure* of the embedding space and performs the task of feature selection. However, in recent TPSR literature this role of $\mathbf{J}$ is not considered to the full extend. Instead, $\mathbf{J}$ is chosen in a way that disregards embedding space structure and semantics as in [9, 3].

In contrast, we want to employ $\mathbf{J}$ to impose constraints or priors on the geometric structure of the embedding space $\mathcal{S}$ which is learned by the TPSR. Unfortunately, Eq. (5) and (6) require that $\mathbf{J}\boldsymbol{P}_{\mathcal{T},\mathcal{H}}$ has full row rank, which imposes a difficult optimization constraint. However, by redefining the subspace transform $\mathbf{Z}$ as

$$\mathbf{Z} = \mathbf{J}\Phi^{\mathcal{T}}\mathbf{R} = \tilde{\mathbf{U}}^\intercal\tilde{\mathbf{J}}\Phi^{\mathcal{T}}\mathbf{R} \quad , \tag{8}$$

we can instead of $\mathbf{J}$ consider a matrix in a similar role, $\tilde{\mathbf{J}}$, that allows for a proper pseudo inverse [8]. To this end, we restrict $\mathbf{J}$ to map to an orthonormal basis using the left singular vectors $\tilde{\mathbf{U}}$ of the product $\tilde{\mathbf{J}}\boldsymbol{P}_{\mathcal{T},\mathcal{H}}$.

As a result, we can modify $\tilde{\mathbf{J}}$ as long as $\tilde{\mathbf{J}}\boldsymbol{P}_{\mathcal{T},\mathcal{H}}$ has full rank and compute $\tilde{\mathbf{U}}$ after the optimization. Instead of the embedding space $\mathcal{S}$, we optimize the embedding space $\tilde{\mathcal{S}}$ generated by the mapping $\tilde{\mathbf{J}}\colon \phi^{\mathcal{T}} \mapsto \tilde{\mathbf{s}}$.

B. Optimization Problem

In order to learn the matrix $\tilde{\mathbf{J}}$ and thus also $\mathbf{J}$, we formulate an optimization problem over the matrix $\tilde{\mathbf{J}}$ of dimension $d \times n_{\mathcal{T}}$. Thereby denotes d the dimensionality of the subspace and $n_{\mathcal{T}}$ the number of features. The work of [11] has $d =$

2 and considers observation features with $n_{\mathcal{T}} = 300$, and all matrix entries are optimization parameters. In our case, features explain sequences of actions and high-dimensional observations. Therefore, the required $n_{\mathcal{T}}$ scales exponentially with the length of the considered tests. To reduce the number of parameters, we parameterize $\tilde{\mathbf{J}}$ by the first m left singular vectors of $\boldsymbol{P}_{\mathcal{T},\mathcal{H}}$, denoted by $\mathbf{U}_{1:m}$.

$$\tilde{\mathbf{J}} = \mathbf{A}\mathbf{U}_{1:m}^{\intercal}, \qquad \mathbf{A} \in \mathbb{R}^{d \times m} \tag{9}$$

The optimization parameters are hence the $d \times m$ entries of the matrix $\mathbf{A}$. We impose our priors on the embedding space by formulating an unconstrained optimization problem with the loss-function $L(D, \mathbf{A})$:

$$\underset{\mathbf{A} \in \mathbb{R}^{m \times d}}{\text{minimize}}\, L(D, \mathbf{A}) \quad . \tag{10}$$

The symbol $D = \{(\phi_t^{\mathcal{T}}, \ell_t)\}_{t=1}^{n}$ denotes a training set of test feature vectors and labels which we require to impose informed priors. Our labels take the form of annotations endowed with a distance metric which assists the underlying planning problem. Usually, loss-functions will operate on the image of test features under $\tilde{\mathbf{J}}$ which we denote by $\tilde{\mathbf{s}}_t = \mathbf{A}\mathbf{U}_{1:m}^{\intercal}\phi_t^{\mathcal{T}} = \tilde{\mathbf{J}}\phi_t^{\mathcal{T}}$.

C. Priors as Loss-functions

In [11], five robotic priors are presented for learning a representation in a *fully observable* setting: *simplicity*, *temporal coherence*. *proportionality*, *causality*, and *repetability*. These priors are derived from the fact that the laws of physics govern the change of the world and the effects of the robot's actions. In this work, we consider the partially observable domain and learn a belief space representation. Below we discuss these priors in the context of our scenario.

The *simplicity* prior prefers representations devoid of redundant information. It is realized in [11] by restricting the representation to few dimensions. We implement this prior by projecting to a low-dimensional subspace and restrict $\mathbf{A} \in \mathbb{R}^{d \times m}$ to a small $d = 5$.

The *temporal coherence* prior requires that states change gradually in time and is implemented in [11] by penalizing distance between successive states. We implement this prior as a loss-function,

$$L_{\text{temporal coherence}} = \mathbb{E}(||\tilde{\mathbf{s}}_t - \tilde{\mathbf{s}}_{t+1})||^2) \quad , \tag{11}$$

where the test at two successive time steps overlap in time.

The *proportionality* prior relates magnitude of an action to magnitude of observed change. This is sensible, if action semantic is context independent. In our example, actions are highly context dependent and we omit this prior (Sec. V-E).

The *causality* prior requires that situations must be dissimilar if different reward is perceived. In our case, reward is not available when the representation is learned as different planning tasks are possible [25].

The *repeatability* prior prefers when similar actions and contexts lead to similar results. This prior is restricted to domains with limited uncertainty as actions in similar (uncertain) believe states can lead to different outcomes. In the navigation domain (see Sec. V), facing the central obstacle and turning can then lead to different observations. The limitation of the last two prior is already stated in [11].

Additionally to the priors above, we introduce *label consistency*. Embedded features should be similar, if their labels are similar, and dissimilar if their labels are dissimilar. This is clearly not always correct in partially observable environments where similar features could be obtain with different labels. However, we want to create a sufficient statistic of history by considering features of tests which are supposed to disambiguate such situations by integrating multiple time steps. Therefore these priors can be permitted:

$$L_{\text{label consistency I}} = \mathbb{E}(e^{(-||\tilde{\mathbf{s}}_{t_1} - \tilde{\mathbf{s}}_{t_1}||^2)}||\ell_{t_1} - \ell_{t_2}||^2) \quad . \tag{12}$$

The value is large if the labels are very different and the embedded features are very similar. This gives incentive to separate the embedded feature vectors. Further, we formulate the loss-function,

$$L_{\text{label consistency II}} = \mathbb{E}(e^{(-||\ell_{t_1} - \ell_{t_2}||^2)}||\tilde{\mathbf{s}}_{t_1} - \tilde{\mathbf{s}}_{t_1}||^2), \tag{13}$$

which results in a large value for pairs of similar labels with dissimilar embedded features, it therefore penalizes inconsistency of the learned representation with the labels.

Besides the priors discussed above, there exist priors specific to the subspace learning method of TPSRs that take the form of model regularization. For example, [14] minimizes the leading eigenvalue and [7] minimizes the sum of the eigenvalues. In the framework that we have described any prior that can be formulated as a geometrical constraint can be included, making it possible to include additional information about the planning task in a flexible manner.

V. EXPERIMENTS

In this section we provide results of the experimental evaluation of our PSRs learned with priors. For this, we consider two partially observable domains with non-linear dynamics—one synthetic task and one with robotic real-world data. We provide a qualitative analysis of the learned P-PSR and show quantitative results that indicate the benefits for planning in this representation as compared to the state-of-the-art baseline. The proficiency for model-based planning is investigated by comparing performance on our P-PSR, the spectral-based PSR, and a memoryless (model-free) agent. For manipulation task data we show how our method handles challenges posted by unbalanced training data.

Before proceeding to describe the experiments, we sketch the kernel-based extension for modeling continuous domains with features proposed by [3] and our reinforcement learning procedure for completeness.

A. Implementation: Training and Planning

We learn the TPSR parameters $\boldsymbol{\beta}_*$, $\boldsymbol{\beta}_\infty$, and $\mathbf{B}_{ao}$ from empirical estimates of the observable matrices introduced in Sec. III-A and III-B. When the domain prohibits direct sampling of execution traces, long sequences are split into

shorter overlapping sequences of fixed length with the suffix-algorithm [29]. The training data therefore consists of fixed-length traces $T = \{tr^{(i)}\}_i$ of which a subset is selected as kernel basis functions for test and history features. Kernel density estimation based on a subset $\mathcal{O}_{\text{center}} = \{o^{(i)}\}_i$ of the observations is used for the indicator function in Eq. (3). For each pair $ao^{(i)} \in \mathcal{A} \times \mathcal{O}_{\text{center}}$, we compute one matrix,

$$\hat{\mathbf{B}}_{ao^{(i)}} = \sum_{tr \in T_a} \left(\mathbf{J}\phi^{\mathcal{T}}_{t+1}(tr) \frac{1}{Z_t} \mathcal{K}(o_t, o^{(i)}) \phi^{\mathcal{H}}_t(tr)^\intercal \right) (\mathbf{J}\boldsymbol{P}_{\mathcal{T},\mathcal{H}})^+ \tag{14}$$

where Z_t is the partition function and $T_a \subset T$ contains all traces with action a at the middle position. Weighting factors cancel out and are omitted. A more detail description is provided in our previous paper [25] and [3].

To close the learning-planning loop, we plan a policy that maps (belief) states to actions $\pi \colon \mathcal{B} \to \mathcal{A}$. The PSR space $\mathcal{B}$ is similar to a traditional belief space as every state $\mathbf{b}(h) \in \mathcal{B}$ represents expected test features. We populate $\mathcal{B}$ by applying Eq. (2) iteratively,

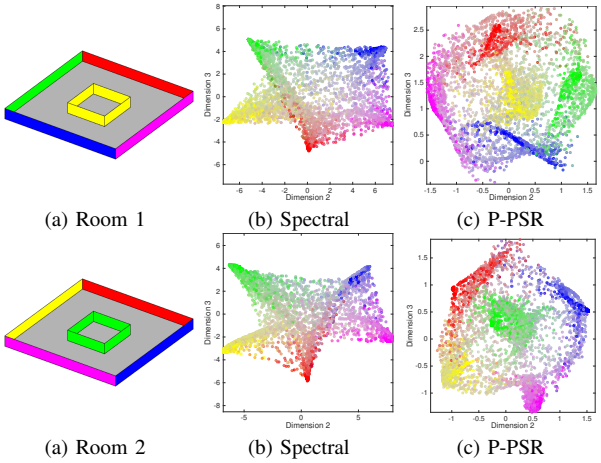
$$\mathbf{b}(h) = \frac{\mathbf{B}_m \mathbf{B}_{m-1} \dots \mathbf{B}_1 \hat{\boldsymbol{\beta}}_*}{\hat{\boldsymbol{\beta}}_\infty^\intercal \mathbf{B}_m \mathbf{B}_{m-1} \dots \mathbf{B}_1 \hat{\boldsymbol{\beta}}_*} \quad , \tag{15}$$

which means filtering (tracking the state) for traces $h = tr^{(i)}$ to get PSR belief points $\mathbf{b}(h)$. In the kernalized formulation, each operator $\mathbf{B}_i$ is a convex combination of the learned operators $\mathbf{B}_j = \sum_i \frac{1}{Z_j} \mathcal{K}(o_j, o^{(i)}) \hat{\mathbf{B}}_{ao^{(i)}}$, according to kernel response. Instead of a defined initial state $\hat{\boldsymbol{\beta}}_0$, we employ an estimated average initial state $\hat{\boldsymbol{\beta}}_*$. This leads to a set of states $B = \{\mathbf{b}(tr^{(1)}), \mathbf{b}(tr^{(2)}), \dots\} \subset \mathcal{B}$. When computing B, we record the traces and for each action save the pair of states. In this way, we encode dynamics by exploiting that similar states are geometrically close neighbors in $\mathcal{B}$. Based on this finite representation we use value iteration on the belief points B and determine a greedy policy.

B. Different Rooms

In this experiment, we compare the learned representation of the spectral learning approach and our prior-based approach by projecting test sequence features into the model space $\mathcal{S}$. The application is a challenging synthetic robot navigation task similar to the one used for representation learning in [24, 3, 11]. In specific, the robot needs to learn a representation of a room by observing images from a simulated camera in an egocentric view.

Experiment domain: A robot moves in a 45×45 unit large room with colored 4 unit high walls (see Fig. 2). Six actions are possible as the product of 0 or 1 unit forward translation and rotation by -15, 0, and +15 degrees. In case where the robot contacts with a wall, it stops, mimicking an elastic collision. Note that the agent only perceives 10-by-10 pixel RGB images of a simulated forward-looking camera with 45 degrees view angle as depicted in Fig. 3. From a single observation it is not possible to discriminate the location in the room. This means that in order to discover its state, the robot needs to integrate observations over time. The



(a) Room 1 (b) Spectral (c) P-PSR

(a) Room 2 (b) Spectral (c) P-PSR

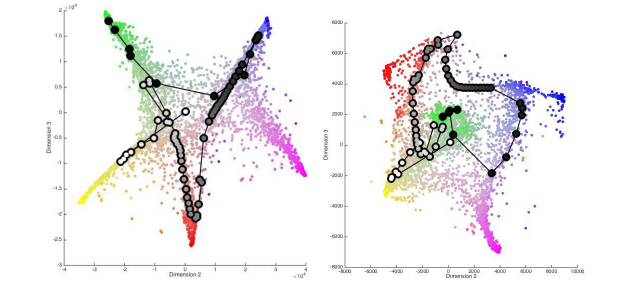
Fig. 2. Visual comparison of the representations learned for two room ((a) and (d)) by the spectral approach ((b) and (e)) and our P-PSR ((c) and (f)). We embed test features for execution traces and color them by the average color of the last camera image. It can be seen that the spectral approach represents the two different environments by the very similar embedding spaces $\mathcal{S}$. Both times, the topology is arranged according to RGB-value similarity. In contrast, our approach captures the geometrical structure of the environment, arranging the features such that the topology of the *environment* is retained.

training data consists of 5000 randomly sampled execution traces of 7 random actions. We use an RBF kernel applied to a "stacked" image (3×300 dimensions) with 2000 basis functions and 500 observation kernel centers. The projection matrix $\mathbf{J}$ is learned with $d = 5$ and $m = 20$ using position and orientation of the robot as labels. Our implementation uses the Broyden-Fletcher-Goldfarb-Shanno Quasi-Newton method with a cubic line search procedure for numerical optimization. We execute 300 iterations which consumes less then 5 min. on a 2.8 GHz Intel Core i7 computer.

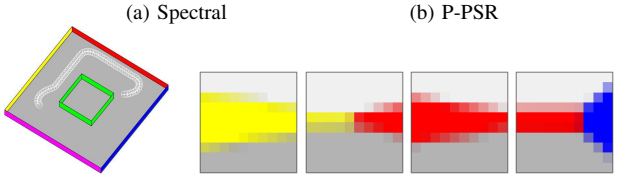
The motivation to perform this experiment comes from [3] where the feature-based spectral TPSR was introduced. It is not trivial to manually define an observation-to-state mapping for this problem. Since the observations are high-dimensional a feature-based representation is required. As such, it is an ideal example to show the benefits of TPSRs. For comparison, we learn such a model with 5 dimensions.

It has been described that the topology of the agent's visual *environment* gets represented in the low-dimensional embedding [3] (see Fig. 2(b)). However, in the spectral approach there is nothing explicitly encouraging this. Rather, the structure of the embedding comes from the structure of the observation space, i.e. the structure of the RGB space. The discover of the room's topology is purely coincidental by the fact that the ordering of the color of the walls coincides with their distance in RGB space. To show this, we altered the layout of the room and exchanged the *blue* with the *magenta* and the *green* with the *yellow* wall (see Fig. 2 (a) & (d)).

As can be seen from the resulting embedding in Fig. 2, the representation learned by the spectral approach is very similar for the two rooms. The *magenta* wall is placed between the *blue* and the *red* wall for both rooms. This is because *magenta* is, in RGB space, a mix of *red* and *blue* and the representation has very little to do with the configuration

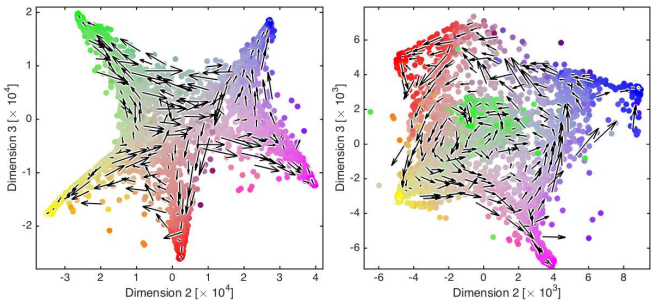


(a) Spectral (b) P-PSR



(c) Robot path (d) Simulated 10×10 pixel RGB camera images

Fig. 3. We embed (track) a set of different robot paths in the virtual environment with simulated camera input and color the belief points by the average color of the last camera image ((a) & (b)). For the long example path (c), the robot needs to derive its state from raw image input only, starting at the position next to the yellow wall. The corresponding trace in the PSR space $\mathcal{B}$ is colored from white to black from start to end and shown in (a) & (b).



(a) Spectral (b) P-PSR

Fig. 4. *Left and middle:* Representation of actions in the PSR belief space. Arrows show the motion of belief points in $\mathcal{B}$ for the move forward and turn left action ((a) and (b)). In the spectral model similar belief points move in different directions, which implies different reward. In the P-PSR model belief point a similar region move in a homogenous manner.

of the walls. In fact, non of the other 2 or 3-dimensional projections encodes the rooms geometric topology. Differently from the spectral method, our approach discovers the underlying structure of the room—due to the priors—and the two embeddings retain the different structure. Further, our representation is capable of placing the inner-wall of the room equally distant from the outer walls reflecting the true topology of the room in a geometric representation. We argue that the representation from our approach is better suited for planning purposes as it reflects the structure of the problem domain and at least is easier to interpret in few dimensions.

C. Belief Space

In the previous section, we inspected the feature embedding space $\mathcal{S}$, which is the image space for test features $\phi^{\mathcal{T}}$. In this section, we investigate the geometric properties of a filtered trace of actions and observations in the PSR belief space $\mathcal{B}$. The PSR belief space is related to the test feature embedding space since the first represents expectations of test features and the latter defines the structure of test features.

1) Tracking a long path: Embedding a long history allows us to understand connectivity in the learned representation. We track the PSR belief state of the trace depicted in Fig. 3(c), starting next to the yellow wall. The tracking result can be seen in Fig. 3(a)&(b).

In the case of the representation learned by the spectral approach, the trace starts in the *gray* area, describing that the average initial state *sees* mostly *gray* pixels. After that the state moves to *yellow*, *green* and *red*. When the robot turns from the *red* wall to the *blue* wall, the trace needs to travel to the opposite side of the PSR belief space to represent states which see mostly *blue* pixels.

In our representation, the initial average state sees a mixture between *green* and *gray* pixels, which relates to the fact that the central obstacle and the floor are visible in about half of the possible robot poses. The trace then travels towards *green* and later to *red* and *blue* in small and regular steps. When the robot turns towards the *blue* wall, the sensor input changes rapidly to *blue* pixels and a larger step can be seen in the belief space. The same phenomena occurs when the robot finally turns towards central obstacle.

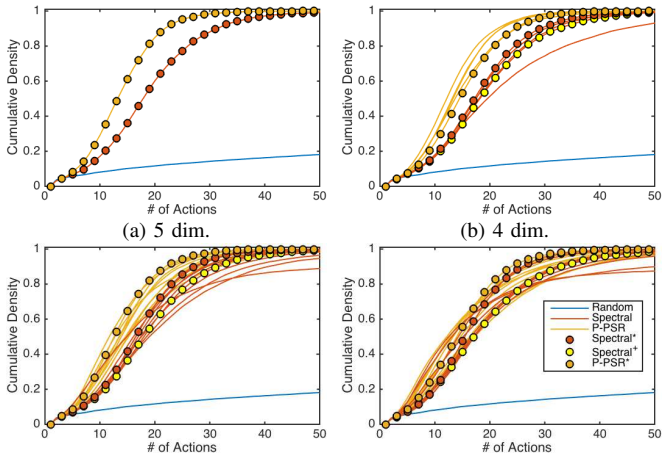
Comparing the two different embeddings we argue that the P-PSR belief space is more suitable for planning and interpretation as it is more temporally coherent not showing such drastic "jumps" as the spectral method.

2) Representation of actions: For planning or policy learning it can be advantageous when the same action has similar effect in neighboring states because it allows to generalize. In Fig. 4 we show the results of applying an action which turns the robot counter-clock wise and moves it forward.

We can observe that in the spectral-based representation all actions are correctly represented, e.g. the states which *see* the *magenta* wall turn towards the *blue* wall. However, the motion of belief points from *yellow* to *magenta* overlaps with the motion from *blue* to *red*. Additionally, we can observe that each of the "corners" exhibits a simultaneous in- and outward motion not far apart. This is semantically correct and relates to the rotational action, however it impairs belief space generalization. Different from that, the P-PSR representation displays a circular motion of belief such that neighboring states are transformed in a similar manner.

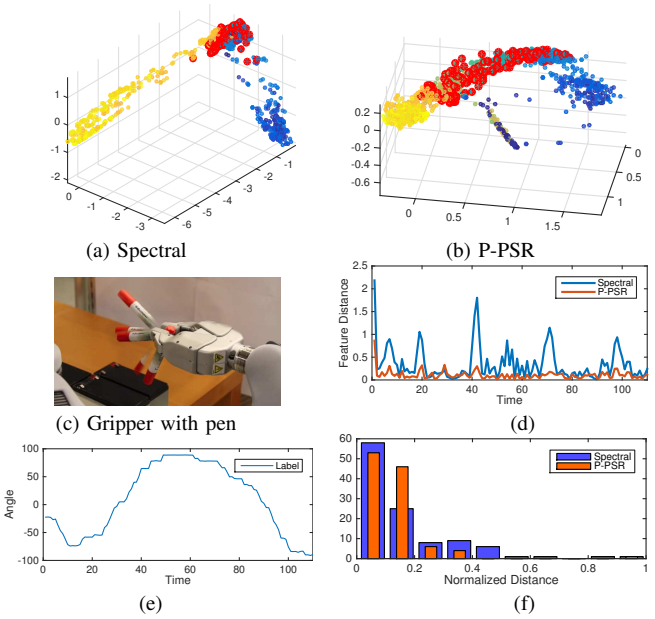
D. Planning in the learned model

Finally, we quantitatively compare how pertaining the learned representations are to subsequent reinforcement learning as this is the main motivation for representation learning. For this, we learn a policy to face the *blue* wall and describe the goal states as seeing an average color with an RGB space distance less than 0.2. The belief points for value iteration are generated by tracking 5000 sample traces for 3 steps. Positive reward (+100) is defined for goal states and wall collisions are penalized (reward -1000). Additionally, we set -100 reward for states that directly face a different wall. As a measure of performance, we compare



(a) 5 dim. (b) 4 dim. (c) 3 dim. (d) 2 dim.

Fig. 5. Analysis of planning performance from 10000 random starting positions to the same target. In (b) to (d) performance on all subsets of the 5 learned dimensions are shown. The random agent (blue) performs worst. In each case the P-PSR models have improved performance by using shorter paths. *P-PSR** and *Spectral** mark the models with best empirical reward and *Spectral*$^{+}$ marks the spectral model using dim. 1 to k.



(a) Spectral (b) P-PSR (c) Gripper with pen (d) (e) (f)

Fig. 6. Learning a representation for rotating a grasped pen with tactile feedback: Spectral method (a) P-PSR method (b). Colors *yellow* to *blue* indicate the angel of the pen. *Red* indicates pen angles between -38 and + 54 degrees. The spectral model represents many different angles at the same place (*red* color) while the P-PSR model spreads out all angles evenly. When tracking a the manipulation with angles shown in (e), the P-PSR representation is more continuous (more smooth) as compared to the spectral model in terms of feature distance (d)&(f).

the number of steps to reach the goal for 10000 randomly sampled starting states as in [3]. Less steps indicate a better performance. Note that the robot never has access to pose, only images.

As most real-world systems are complex, we often cannot learn an exact low-dimensional PSR model [14] and therefor have to violate the sufficiency assumption of the spectral T-PSR learning algorithm. To see the effects of learning approximate PSR models, we investigate planning performance on all subsets of the 5 model dimensions for both approaches. Better planning with less dimensions indicates better approximation capabilities and consequently superior suitability for real-world application.

In Fig. 5 we can see that the baseline agent that performs random actions results in much longer action sequences than both of the model-based agents. The spectral models result in an substantial improvement over the baseline (as also reported by [3]) and the P-PSR agents again improve upon this result. In detail, for 2 dim. the P-PSR model with best empirical reward (-14.25) uses dim. 2 and 3, the spectral model with best reward (-24.51) uses dim. 2 and 5 while the model with dimensions 1 and 2 yields empirical reward -60.03. Overall best empirical reward is -5.27 by a 3-dim. P-PSR model compared to -29.08 for the 3-dim. spectral model. For 4 dim., the best empirical rewards are -13.94 (P-PSR) and -40.45 (spectral) and for 5 dim. -13.84 (P-PSR) and -52.22 (spectral). In no case were the first k dimensions the best spectral model, which is counter intuitive to the algorithm's spectral paradigm.

We assume that this improvement is due to a better approximate representation since the policy learning algorithm exploits belief space similarity for generalization.

E. Biased Sample Set

When learning representations for robotic systems that interact with the world collecting training examples can be very time consuming, expensive, or at least tedious. Additionally, it can be difficult to collect an unbiased set of examples or to design an automatic exploration policy that generates a uniformly distributed and representative set.

In this experiment, we consider data from the robotic manipulation domain of our previous work [25]. The task is to change the orientation of a grasped pen by pushing it against a table edge in different ways as illustrated in Fig. 6(c). The robot can only access pressure readings from its fingertips and the two most extreme regions of angles (+90 and -90 degrees) have identical readings. In all positions of the pen, at least one of the six actions can affect its pose, but when the pen points straight forward, all push actions result in change. This generates a bias in the sample set in such that certain pen angles are less frequently represented.

We learn a 5 dimensional representation with $m = 20$ and all our objective functions using the visually obtained pen angles labels. For the kernelization we use 342 basis function and 12 observation kernel centers. The 1312 training examples are obtained with the suffix algorithm [29].

As we can observe in Fig. 6(a), the spectral method places many different pen states (angles of -38 to +54 degrees) into the region indicated by red color. In contrast, the P-PSR spreads out all angles evenly. The small line in the center represents all traces which remain indistinguishable because they never leave the region where the observations are aliased. In accordance with the priors, these traces are places in the middle between the states to which they are most similar.

VI. CONCLUSIONS

We have presented an approach that extends PSRs to learn representations that are meaningful for planning. Our approach incorporates prior information about the planning task during the learning phase encouraging the model to retain information that are relevant for the task. We presented experimental results showing the benefits of our approach using quantitative and qualitative measures. In specific, we show how our approach is capable of learning in scenarios where the training data-set is biased and when only a subset of the variations in the observation features are relevant. The generality of our framework allows for future development including additional priors thereby significantly extending the domains where PSRs are applicable.

REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent. "Representation learning: A review and new perspectives". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.8 (2013), pp. 1798–1828.

[2] B. Boots and G. J. Gordon. "An Online Spectral Learning Algorithm for Partially Observable Nonlinear Dynamical Systems." In: *AAAI*. 2011.

[3] B. Boots, S. Siddiqi, and G. Gordon. "Closing the Learning Planning Loop with Predictive State Representations". In: *IJRR* 30 (2011), pp. 954–956.

[4] B. Boots, A. Gretton, and G. J. Gordon. "Hilbert Space Embeddings of Predictive State Representations". In: *UAI*. 2013.

[5] M. Bowling, A. Ghodsi, and D. Wilkinson. "Action respecting embedding". In: *ICML*. ACM. 2005.

[6] D. Ernst, P. Geurts, and L. Wehenkel. "Tree-based batch mode reinforcement learning". In: *JMLR*. 2005, pp. 503–556.

[7] H. Glaude, O. Pietquin, and C. Enderli. "Subspace identification for predictive state representation by nuclear norm minimization". In: *IEE ADPRL*. 2014.

[8] W. Hamilton, M. M. Fard, and J. Pineau. "Efficient Learning and Planning with Compressed Predictive States". In: *JMLR* 15 (2014), pp. 3395–3439.

[9] W. L. Hamilton, M. M. Fard, and J. Pineau. "Modelling sparse dynamical systems with compressed predictive state representations". In: *ICML*. 2013.

[10] M. T. Izadi and D. Precup. "Point-based planning for predictive state representations". In: *Advances in Artificial Intelligence*. Springer, 2008, pp. 126–137.

[11] R. Jonschkowski and O. Brock. "State Representation Learning in Robotics: Using Prior Knowledge about Physical Interaction". In: *RSS*. Berkeley, USA, 2014.

[12] L. P. Kaebling. *"Keynote Lecture: Robot Intelligence"*. AAAI Conference on Artificial Intelligence. Atlanta, Georgia, USA. 2010.

[13] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. "Planning and acting in partially observable stochastic domains". In: *Artificial intelligence* 101.1 (1998), pp. 99–134.

[14] A. Kulesza, N. Jiang, and S. Singh. "Spectral Learning of Predictive State Representations with Insufficient Statistics". In: *AAAI*. 2015.

[15] S. Lange, M. Riedmiller, and A Voigtlander. "Autonomous reinforcement learning on raw visual input data in a real world application". In: *IJCNN*. IEEE. 2012, pp. 1–8.

[16] M. Littman, R. Sutton, and S. Singh. "Predictive Representations of State". In: *NIPS*. 2002, pp. 1555–1561.

[17] D. Nguyen-Tuong and J. Peters. "Model learning for robot control: a survey". In: *Cognitive processing* 12.4 (2011), pp. 319–340.

[18] S. C. Ong, Y. Grinberg, and J. Pineau. "Goal-directed online learning of predictive models". In: *Recent Advances in Reinforcement Learning*. Springer, 2012, pp. 18–29.

[19] J. Pearl. *Causality: models, reasoning and inference*. Vol. 29. Cambridge University Press, 2000.

[20] M. Riedmiller. "Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method". In: *ECML*. Springer, 2005, pp. 317–328.

[21] M. Rosencrantz, G. Gordon, and S. Thrun. "Learning low dimensional predictive representations". In: *ICML*. ACM. 2004, p. 88.

[22] H. Shatkay and L. P. Kaelbling. "Learning geometrically-constrained hidden markov models for robot navigation: Bridging the topological-geometrical gap". In: *Journal of Artificial Intelligence Research* 16.1 (2002), pp. 167–207.

[23] S. Singh, M. R. James, and M. R. Rudary. "Predictive state representations: A new theory for modeling dynamical systems". In: *UAI*. 2004, pp. 512–519.

[24] N. Sprague. "Predictive Projections." In: *21st International Joint Conference on Artificial Intelligence (IJCAI)*. 2009, pp. 1223–1229.

[25] J. Stork, C. Ek, Y. Bekiroglu, and D. K. "Learning Predictive State Representation for In-Hand Manipulation". In: *IEEE ICRA*. Seattle, USA, 2015.

[26] S. Thrun. "Monte Carlo POMDPs." In: *NIPS*. Vol. 12. 1999, pp. 1064–1070.

[27] D. Wingate and S. Singh. "On discovery and learning of models with predictive representations of state for agents with continuous actions and observations". In: *International joint conference on Autonomous agents and multiagent systems*. ACM. 2007, p. 187.

[28] L. Wiskott and T. J. Sejnowski. "Slow feature analysis: Unsupervised learning of invariances". In: *Neural computation* 14.4 (2002), pp. 715–770.

[29] B. Wolfe, M. R. James, and S. Singh. "Learning predictive state representations in dynamical systems without reset". In: *ICML*. ACM. 2005, pp. 980–987.

[30] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell. "Distance Metric Learning with Application to Clustering with Side-Information." In: *Advances in Neural Information Processing Systems*. 2002, pp. 505–512.