Learning Grounded Relational Symbols from Continuous Data for Abstract Reasoning

Nikolay Jetchev, Tobias Lang, Marc Toussaint

***Abstract*— Learning from experience how to manipulate an environment in a goal-directed manner is one of the central challenges in research on autonomous robots. In the case of object manipulation, efficient learning and planning should exploit the underlying relational structure of manipulation problems and combine geometric state descriptions with abstract symbolic representations. When appropriate symbols are not predefined they need to be learned from geometric data. In this paper we present an approach for learning symbolic relational abstractions of geometric features *such that* these symbols enable a robot to learn abstract transition models and to use them for goal-directed planning of motor primitive sequences. This is framed as an optimization problem, where a loss function evaluates how predictive the learned symbols are for the effects of given motor primitives as well as for reward. The approach is embedded in a full-fledged symbolic relational model-based reinforcement learning setting, where both the symbols as well as the abstract transition and reward models are learned from experience. We quantitatively compare the approach to simpler baselines in an object manipulation task and demonstrate it on a real-world robot.**

I. Introduction

How can autonomous robots learn to manipulate their environment in a goal-directed manner? In natural environments composed of objects, a robot has to reason on a geometric as well as on an abstract level to learn and plan sequences of motor primitives [1]. The geometric level concerns mostly sub-symbolic perception and motion generation where machine learning and robotics research was greatly successful in the last decades. In contrast, to efficiently determine an appropriate sequence of motions from a given set of motor primitives requires learning and reasoning on an abstract level. Symbolic relational representations are a promising approach to abstract reasoning. They reflect the structure inherent in object manipulation tasks in terms of abstract properties and relations between objects; and they generalize over object identities to new situations with previously unseen objects. For this reason, symbolic representations have been investigated in machine learning and artificial intelligence from their beginnings.

It is a key open challenge, however, how sub-symbolic and symbolic learning and reasoning approaches can be combined. To apply symbolic approaches in the physical world, one needs to answer the fundamental questions: *"Where do the symbols come from?"* and *"How are the given symbols grounded in the physical world?"* This is known as the symbol grounding problem in psychology and cognitive science [2]. In our view, a core challenge to answer these questions and to progress in the research on autonomous agents is to leverage learning methods to bridge the gap between continuous state descriptions and symbolic features. Learning should bootstrap from continuous state descriptions symbolic abstractions which are empirically useful as a basis for symbolic learning and reasoning.

The authors are with the Machine Learning and Robotics Lab, FU Berlin, Arnimallee 7, 14195 Berlin, Germany {nikolay.jetchev,tobias.lang,marc.toussaint}@fu-berlin.de

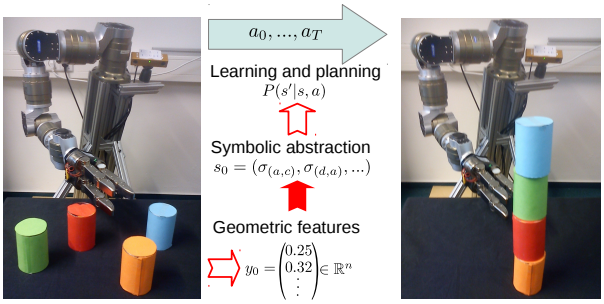


Fig. 1. An illustration of our approach: using continuous geometric features y from its sensors the robot learns symbols σ and uses them in a symbolic state description s for planning an action sequence maximizing the reward. Our major contribution is the learning step indicated by the filled red arrow which takes the whole robot performance into account.

In this paper, we approach the problem by assuming that the robot has a fixed set of motor primitives (corresponding to symbolic actions) and perceives continuous geometric properties and relations between objects. To be able to choose and sequence appropriate actions, the robot has to learn abstract state symbols which capture the general patterns underlying the effects of its motor primitives on those objects. We formalize this symbol learning problem in a reinforcement learning (RL) framework where the goal of the robot is to find motor primitive sequences which maximize its future rewards. The suitability of learned state symbols gets thereby directly linked to the robot's ability to perform goal-directed behavior.

We define a symbol as a relational predicate (initially without semantics) paired with a mapping (classifier) from the geometric features to a true/false value of the predicate. Learning symbols then translates to learning appropriate classifiers. We propose a loss function which favors symbols that allow to consistently and discriminately predict the effects of motor primitives and rewards on an abstract level. We integrate this symbol learning procedure in a full-fledged symbolic relational model-based RL approach: the symbols (associated classifiers) as well as the abstract transition and reward models are all learned from experience. Experiments in a robot manipulation scenario, both in simulation and on a real robot, show that this integrated approach allows an

autonomous robot to find a task-relevant structure in its state observations which it can exploit for efficient planning of sequences of motor primitives. Figure 1 illustrates our work.

In the next section we discuss related and previous work on symbol learning. Section III summarizes background on relational reinforcement learning, on which our approach in parts builds. Section IV then introduces our specific approach to formalizing the symbol learning problem before, in Section V, we define the loss function for learning the symbol's classifiers. In Section VI we demonstrate the approach on a robot manipulation domain in simulation and in the real world.

II. Related work

The origin of symbols that humans seem to use to structure their life has been a subject of study for long [2]. In a language acquisition context, linguists have researched spatial predicate extraction from natural language instructions in a supervised way. [3] learn the meaning of words like "past" and "to" by analyzing motion trajectories and their descriptions from a teacher. Approaches to extract rules and non-relational symbols from a neural network trained for supervised learning have been investigated in very simple computer simulation scenarios [4], [5]. [6] showed that it is important that grounded symbols are coupled and proposed predictive, correlation and contrastive criteria for learning. In a biological context, [7] showed that apes can learn symbols from examples provided by a teacher who provides significant guidance. In contrast to our work, all these approaches do not focus on learning symbols *to enable* an autonomous robot to act in a goal-directed way in a world with objects.

From their early beginnings, research in artificial intelligence and cognitive science has investigated modeling reasoning and learning of autonomous agents based on symbolic relational representations of the world [8], not without significant objections [9]. Over the last decade, symbolic relational approaches have received new enthusiasm with the advent of machine learning techniques in AI and cognitive science [10], [11]. However, the critique of "where the symbols come from" remains. Feldman [12] provides a formal computational model of the conditions when a symbolic representation can capture the variability of a continuous environment. But how symbols can be learned and used for reasoning has, to our knowledge, not been shown. Generally, learning symbols in the context of actually working and acting systems has seen only little progress. [13] investigated how a real robot can learn to manipulate articulated objects using a grounded relational representation, but did not learn the kinematic symbols themselves. [14] discusses that a robot needs to learn structured symbols according to experience which reflect spatiotemporal correlations in its sensory predictions, motor signals, and internal variables (sensorimotor invariances in its actions). [15] present a study of a real robot which learns non-relational symbols by means of neural networks for manipulation tasks. To our knowledge, our approach is the first attempt to learn symbols to enable model-based relational reinforcement learning.

III. Background on relational RL

In this section we review background on relational (esp. model-based) reinforcement learning [10]—an area of research which so far has been applied using predefined symbolic representations. Consider an environment with M objects. Learning transition models (the probabilistic effects of motor primitives or actions) directly on a geometric representation is in general hard. Natural environments are profoundly structured in terms of their compositionality by objects. A standard approach to describe the state of multiple objects is in terms of a set of unary and binary predicates $\mathcal{P}$. For instance, given M objects, K unary predicates and L binary predicates, the *propositional state description* $s \in \{0,1\}^\nu$ is given by the $\nu = KM + LM^2$ truth values for the instantiations $p_k(o_m)_{k=1:K,m=1:M}$ and $p_l(o_m, o_{m'})_{l=1:L,m=1:M,m'=1:M}$ of all predicates with all objects (in all permutations)—the propositional state space is exponential in the number of objects.

Relational representations generalize over object identities and thereby reflect the prior assumption that the effects of actions should not depend on the explicit identity of an object, but only on its properties and relations to other objects. Relational reinforcement learning developed methods either for learning relational value functions [10] or learning compact relational transition models $P(s' \mid s, a)$ [16] and using them in a model-based RL setting for planning or further exploration. This generalization greatly reduces the amount of data necessary for an agent to find a good policy and transition model in comparison to propositional learning [17]. In our approach we will learn a transition model (based on learned symbols) in the form of noisy probabilistic relational rules [16] from concrete experience in the format $\mathcal{D} = \{(s_t, a_t, s_{t+1})\}_t$, where s_t, s_{t+1} are propositional state descriptions before and after the application of an action a_t. We use the planning method PRADA [17] based on the learned rules to find action sequences which maximize the expected reward.

IV. Symbol learning problem formulation

Consider an environment with M objects $\mathcal{O} = \{o_1, \ldots, o_M\}$ in state x. We assume that, given the state x, the robot has access to a geometric state descriptor y comprising features $y_i \in \mathbb{R}^n$ for each object o_i and features $y_{ij} \in \mathbb{R}^{n'}$ for each pair o_i, o_j of objects. For instance, y_i may represent the position and size of o_i as computed from the robot's vision system; while a basic example for pair-wise features is $y_{ij} = y_i - y_j$. The robot can execute motor primitives a of different types $\mathcal{A}$ on target objects $o \in \mathcal{O}$; e.g., grab objects and put them on other objects. The execution of motor primitives changes the configuration and thus observed geometric features y of one or more objects.

We are interested in relational symbols to describe the effects of motor primitives. We define a relational symbol $\sigma = (p, f)$ as a predicate p together with its grounding f. The predicate on its own (without the grounding f) is without semantics. A grounding f is a classifier that decides for all possible permutations $\mathbf{o}$ of the objects (for instance, in the case of a binary predicate for all pairs (o_i, o_j),

$o_i, o_j \in \mathcal{O}$) whether the corresponding instantiation $p(\mathbf{o})$ of the predicate holds given the geometric features $\{y_i, y_{ij}\}$ for $\mathbf{o}$. For instance, we calculate the grounding of a unary predicate instantiated with a single object o_i from its features y_i by $f_1 : \mathbb{R}^n \to \{0,1\}$ and similarly for binary predicates for a pair o_i, o_j with features y_{ij} by $f_2 : \mathbb{R}^{n'} \to \{0,1\}$. The predicates together with their binary classifiers form the set of relational symbols Σ. The symbolic relational state $s \in \{0,1\}^\nu$ is a binary vector that represents whether specific predicate instantiations can be successfully grounded in state x: for each symbol $\sigma \in \Sigma$ of arity b, there is an entry in s for each permutation $\mathbf{o}$ of length b of the objects in $\mathcal{O}$. The entry corresponds to the truth value of $p_\sigma(\mathbf{o})$ which is calculated by the classifier f_σ from the geometric features $\{y_i, y_{ij}\}$ for $\mathbf{o}$. A set of symbols Σ thereby defines the state abstraction function $\phi(x; \Sigma) \to s$ which takes a physical world state x and abstracts it to the relational symbolic state s.

The robot receives a continuous reward signal $r \in \mathbb{R}$ in each state. It interacts with its environment in discrete time-steps, resulting in a sequence $x_0, r_0, a_0, x_1, r_1, a_1, \ldots$. The task of the robot is to maximize its rewards $R = \sum_{t=0}^{T} \gamma^t r_t$ for discounting $0 < \gamma < 1$. Our approach is to learn symbols Σ that allow to abstract x_t to its symbolic relational representation s_t and then to reason in the abstract representation about motor primitives. Hence, the problem can be summarized as *learning classifiers f_σ that define a set of symbols $\sigma \in \Sigma$ such that model-based RL using these symbols will be successful in maximizing the expected future reward.* Both, the set of symbols as well as the symbolic transition and reward models, have to be learned from experiences $\mathcal{D} = \{(y_t, r_t, a_t, y_{t+1})\}$. This problem formulation frames symbol learning as part of an overall reinforcement learning problem—in contrast to symbol learning in a supervised fashion from a teacher.

V. Learning the symbol groundings f_σ

We define a loss function $L(\Sigma; \mathcal{D})$ over data $\mathcal{D}$ to learn symbols $\sigma \in \Sigma$, that is, to find classifiers f_σ that minimize this loss. The loss function needs to reflect our conception of what good symbols are. As discussed in the previous section, the goal of symbol learning is to give the robot the ability to **plan** efficiently to high-reward states in the physical world. Hence, we want symbols to (i) allow us to learn a **transition model** which one can use to predict the structural effects of executing motor primitives and discriminate between predecessor and successor states; (ii) allow us to learn a **reward model** to predict how the reward values (defining tasks) are related to such structural changes; and (iii) be **compact** so as to allow for efficient reasoning. We define the loss function

$$L(\Sigma; \mathcal{D}) = L_{\text{transition}}(\Sigma; \mathcal{D}) + L_{\text{reward}}(\Sigma; \mathcal{D}) + \|\Sigma\|^2 \,. \quad (1)$$

Learning symbols works by minimizing this loss. The next subsections will explain in detail its three terms. As a sidenote, in linear value function approximation the Bellman error can similarly be decomposed in reward and prediction terms [18].

A. Learning a transition model via $L_{transition}$

Recall that $s_t \in \{0,1\}^\nu$ denotes the truth values of the predicate instantiations in a world state x_t for the given symbols Σ and objects $\mathcal{O}$. A model $\mathcal{T}$ can be learned by abstracting the experiences $\mathcal{D} = \{(y_t, r_t, a_t, y_{t+1})\}$ to their symbolic representation $\mathcal{D}^\Sigma = \{(s_t, r_t, a_t, s_{t+1})\}$ based on the symbols Σ. We define a loss term $L_{\text{transition}}$ which rewards symbols that are suitable for transition model learning as

$$L_{\text{transition}}(\Sigma; \mathcal{D}) = \sum_{(s_t, r_t, a_t, s_{t+1}) \in \mathcal{D}^\Sigma} \|s_t - s_{t+1}\|_1 \quad (2)$$
$$- \sum_{(s_t, r_t, a_t, s_{t+1}) \in \mathcal{D}^\Sigma} \log P(s_{t+1} \mid s_t, a_t; \mathcal{T})$$

where $P(s' \mid s, a; \mathcal{T})$ is the transition probability according to the potentially stochastic transition model $\mathcal{T}$ that is estimated from $\mathcal{D}$ as well. This loss function incorporates two different terms which we describe in the following.

The first term in Eq. (3) rewards *effect discrimination*. This is defined by the Hamming distance $\|s_t - s_{t+1}\|_1$ between state pairs s_t, s_{t+1}. In the data $\mathcal{D}$, motor primitives a_t always change object configurations in the physical world, that is $y_{t,i} \neq y_{t+1,i}$ for at least one $o_i \in \mathcal{O}$. Learned symbols shall reflect these changes to account for the structure inherent in the world and the motor primitives.

The second term in Eq. (3) rewards *prediction accuracy*. This is defined by the data-likelihood of the successor states in $\mathcal{D}^\Sigma$ given the current states and actions according to the learned symbolic relational transition model $\mathcal{T}$. This rewards symbols which allow to predict successor states accurately. Good relational symbols are not only those whose truth values reflect the changes according to motor primitives, but also symbols whose truth values identify relevant objects and define contexts for the type of changes according to the motor primitives.

In principle, any suitable relational method could be used to learn the transition model, such as advanced relational classification techniques and probabilistic relational rules. In our experiments, we use a basic relational nearest neighbor (NN) predictor. We use this efficient technique since we need to estimate $\mathcal{T}$ frequently in every step of our overall optimization procedure (described in Section VI) for Σ. For a given state-action pair s, a, this NN predictor finds the nearest neighbor $\hat{s}_t, \hat{a}_t$ in the data $\mathcal{D}^\Sigma$ and predicts the according successor state $\hat{s}_{t+1}$. Relational generalization over the objects in s, a is achieved by using a permutation-invariant kernel that is described in the Appendix.

B. Learning a reward model via L_{reward}

It is crucial for planning that learned symbols allow to determine state rewards (which define the robot's task). This is achieved by a relational regressor $\mathcal{R} : s \to \mathbb{R}$ and motivates the loss term

$$L_{\text{reward}}(\Sigma; \mathcal{D}) = - \sum_{(s_t, r_t, a_t, s_{t+1}) \in \mathcal{D}^\Sigma} \log P(r_t \mid s_t; \mathcal{R}) \,. \quad (3)$$

where $P(r_t \mid s_t; \mathcal{R})$ is the probability of reward r_t in s_t according to the potentially stochastic reward model $\mathcal{R}$. This

loss reflects how well learned symbols Σ can be used to solve the relational regression problem of predicting the reward r_t from the symbolic state representation s_t.

In our experiments, we again use a relational NN regressor for $\mathcal{R}$. For a given state s, this regressor looks through the data $\mathcal{D}^\Sigma$ for the nearest neighbor $\hat{s}_t$ and predicts the according reward $\hat{r}_t$. As above, relational generalization is achieved by the permutation-invariant kernel described in the Appendix.

C. Regularizing symbols Σ

The last term in the loss function in Eq. (1), $\|\Sigma\|^2$, regularizes the learned symbols Σ. This regularization can be used to formalize different requirements on symbols. Our idea is that we want grounded symbols to represent distinct geometric structures within physical states. This is advantageous for model selection when the number of symbols is unknown in advance: we can start learning with multiple candidate symbols; if a symbol is not needed to represent abstract state structures, it gets pruned from the model due to regularization.

To specify the regularization we need to consider a concrete function class for f_σ to ground symbols. We use radial basis function (RBF) classifiers which are defined by a parameter vector $w = (w_c, w_s)$ consisting of a center and a standard deviation. A symbol $\sigma = (p, w)$ is then defined by the predicate p and the RBF parameters w. For a binary symbol, for instance, we calculate the truth value of $p(o_i, o_j)$ from the activation function

$$g(y_{ij}; w) = \exp(-(y_{ij} - w_c)^\top \operatorname{diag}(w_s^2)^{-1}(y_{ij} - w_c)) \tag{4}$$

based on the pair-wise geometric feature vector y_{ij} of the objects o_i and o_j . If $g(y_{ij}; w) > 0.5$, then $p(o_i, o_j)$ is true. The regularization now applies to the parameterizations $w = (w_c, w_s)$. We focus regularization on the widths w_s which control the activation areas of symbols defined by $g(y) > 0.5$. This regularization leads to a trade-off between predictive power and small activation areas.

VI. Experimental evaluation

We investigated whether (a) our approach allows to learn symbols from continuous geometric data, (b) the resulting symbols are well grounded in the physical world, and (c) the learned symbols enable planning on the symbolic level leading the robot to higher rewards. For learning (approximately) optimal symbols Σ^*, that is the parameters w^* for the RBF classifiers f_σ in our case, we use a stochastic search algorithm (simulated annealing). To speed up convergence we used an operator which swaps complete symbols (w_c, w_s) at once between candidate solutions. About 10^5 loss function evaluations were required to reach optima in the different setups.

To evaluate our approach on two different tasks, manipulating cubes and balls on a table, and playing bowling, and performed a qualitative real-world experiment. The accompanying video includes a real-world experiment as well as a video of our bowling simulation. For each task, we collected traces $\Delta = \{\langle y_t^i, a_t^i, r_t^i \rangle\}_{i=1,t=1}^{n,T}$ of n length-T-sequences of random actions with random start states, i.e. nT data points. These traces were used by the compared methods for learning appropriate models and symbols for planning.

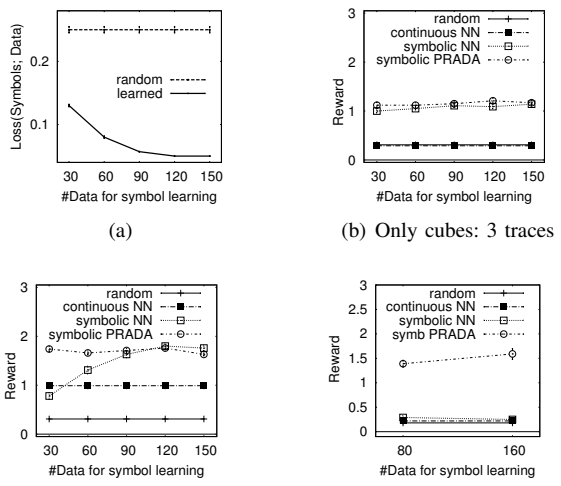
A. Compared methods

We compared four different methods based on either continuous or symbolic representations. RANDOM performs random actions. CONTINUOUS-NN (nearest neighbor) is a baseline that exploits the available data in a relational way but without using a symbolic representation. Geometric states y_t^i within these traces are assigned values $V(y_t^i) = \sum_{d=0}^{3} \gamma^d r_{t+d}^i$ with a discount factor $\gamma = 0.8$. (The y actually denotes the set of geometric features $y = \{y_i, y_{ij}\}$. In this method, y_i are treated as states.) We set $r_{t+d} := r_T$ for $t + d > T$. Given a new geometric state y the robot chooses an action based on nearest neighbor retrieval from these traces: the closest states to y in Δ are retrieved based on the relational permutation-invariant similarity kernel $k(y, y')$ in continuous space (defined in the Appendix). Then the action a^* of the nearest neighbor y^* with the largest value $V(y^*)$ is chosen, where the action arguments are assigned according to the permutation-invariant kernel. SYMBOLIC-NN is equal to CONTINUOUS-NN except that states in the traces are represented abstractly using the *learned* grounded symbols Σ and the permutation-invariant similarity kernel $k(s, s')$ in $\mathcal{S}$ is used. The trace data $\Delta^\Sigma = \{\langle s_t^i, a_t^i, r_t^i \rangle\}_{i=1,t=1}^{n,T}$ corresponds to paths through the abstract symbolic state space. SYMBOLIC-PRADA is a relational planner which uses learned noisy probabilistic relational rules to find appropriate symbolic actions (see Sec. III). Rules are learned from the learned symbolic abstraction Δ^Σ of the experiences—the same data set used for SYMBOLIC-NN. To evaluate action sequences, PRADA requires a symbolic description of the rewards. We approximate the reward function with a simple symbolic regression using relational decision trees with the learned symbols.

We devised these methods to allow for performance comparison w.r.t. different aspects of the representations used. Both CONTINUOUS-NN and SYMBOLIC-NN are relational in the sense that they abstract over object identities via the permutation-invariant kernel. Only SYMBOLIC-NN and SYMBOLIC-PRADA use the learned grounded symbols, and only SYMBOLIC-PRADA represents a full-blown model-based relational RL approach. With this we aim to separate out the effects of a relational representation and the use of learned symbols.

B. Simulated robot manipulation scenario

In our first scenario, a physically simulated robot manipulates balls and cubes on a table (Fig. 3(a)). The robot can execute two motor primitives which are affected by noise: it can grab an object or open its hand over some other object; $\mathcal{A} = \{closeHandAround(X), openHandOver(X)\}$. After grabbing, the robot always retracts to a neutral "home" position. The object observations of the robot are described by continuous feature vectors $y_i \in \mathbb{R}^4$ comprising the 3-dimensional position of the center and the size of object o_i.



(a) (b) Only cubes: 3 traces

(c) Only cubes: 9 traces (d) Cubes and balls

Fig. 2. *Simulated robot manipulation scenario*. (a) Symbol learning loss of randomly defined and learned symbols on 400 test states. (b) / (c) Results for scenario with cubes based on model learning with 3 / 9 traces. The reward for performing no actions is 0. (d) Results for cubes and balls.

The task is to build towers. This is defined by means of a reward function: the average discounted continuous height of cubes over 10 time-steps.

In our first experiment there were 7 cubes of two distinct sizes on the table. We learned one unary predicate $u(X)$ and one binary predicate $b(X, Y)$. We tried learning more symbols, but the regularization term of the loss function would usually eliminate additional symbols. Our results are shown in Figs. 2(a), 2(b) and 2(c). We learned symbols based on increasing numbers of data in $\mathcal{D}$, as given on the x-axes of the diagrams. Fig. 2(a) shows how the validated loss of the learned symbols improves with an increasing number of training data. For learning models (traces for the NN approaches and the probabilistic relational rules for SYMBOLIC-PRADA), we used independent data sets of size 30 ($n = 3$ traces with $T = 10$) (Fig. 2(b)) or 90 ($n = 9$ traces) (Fig. 2(c)). Each combination of learned symbols and models was evaluated in three new starting situations where all objects lay on the table. Thus, the initial reward was 0. The hypothetically optimal reward in the corresponding deterministic world was 2.963. To get statistics, we performed 90 runs with different data split into independent sets for symbol and model learning and different random seeds. RANDOM is independent of symbol and model learning; its performance is thus always the same in Figs. 2(b) and 2(c). CONTINUOUS-NN performs better than RANDOM with a sufficiently large number of data for NN retrieval of actions. The symbolic approaches show superior performance even with limited data for model learning (Fig. 2(b)). Due to the learned symbolic abstraction, SYMBOLIC-NN performs better with more data for NN retrieval (Fig. 2(c)) as long as the learned symbols are sufficiently accurate. In contrast, SYMBOLIC-PRADA is less sensitive to noisy or less accurately learned symbols and outperforms all other methods. This is due to the following reason: in case of less accurately grounded symbols, symbolic structures sometimes get overlooked in a state x_t^i. For instance, the unary symbol $u(X)$ with the canonical meaning X in hand may not be detected in x_t^i although an object is held in hand in x_t^i. This is problematic for nearest neighbor detection. In contrast, SYMBOLIC-PRADA is not bound to NN retrieval in traces, but learns rules from breaking the trace data into individual experiences. Individual faulty symbolic state abstractions are handled by SYMBOLIC-PRADA since the underlying probabilistic relational rules cope with noisy outcomes of actions and identify the typical outcomes of actions as required for effective planning.

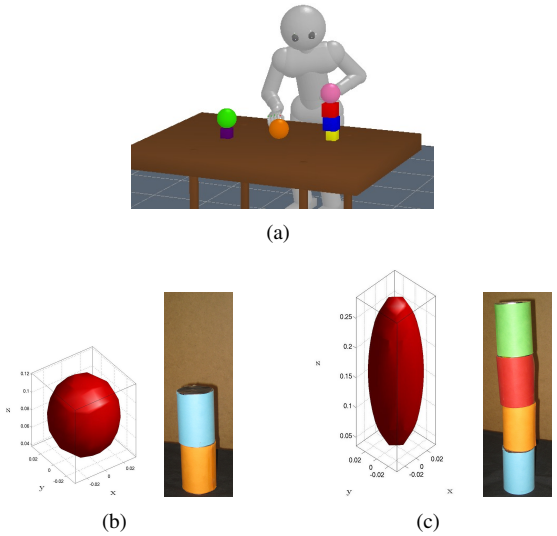
In a second experiment, we investigated the four methods in a scenario extended by balls (in addition to cubes; the total number of objects was seven as above). Here, we learned two unary and one binary symbol with our learning approach. Our results for 80 and 160 data for symbol learning are presented in Fig. 2(d). We used $n = 8$ traces of $T = 10$ actions to learn models. The results confirm the previous findings: the symbolic methods show superior performance. The advantage of SYMBOLIC-PRADA is more significant here since the NN approaches would require much more traces for similar performance due to the increased complexity of the situations.

Overall the results show the great potential of using learned relational symbols for state abstraction in combination with sophisticated relational planning techniques. We also examined briefly the setup of using point cloud data from a Kinect sensor. For object features we used the mean of the point cloud and the first 7 PCA components, hence $y \in \mathbb{R}^{10}$. We were able to learn a symbol corresponding to a spherical shape with our learning method which can be used to predict when an object cannot be stacked.

Figure 3 illustrates what was learned as a binary predicate $b(X, Y)$ in two exemplary runs. In the first run, it can be interpreted as representing that X is directly on top of Y (specifically with a maximum difference in height corresponding to the largest object size); in the second run, that X has similar x/y coordinates as Y but may differ in height—that X is in the tower stacked above Y. In almost all of our runs, the learned unary predicate $u(X)$ could be interpreted as denoting that X is being held in the robot's hand. In a few cases, $u(x)$ was learned to denote that an object is reachable. In the experiments with balls, an additional unary symbol learned to distinguish between balls and cubes by means of the object size (which was different for balls and cubes) since this is important in the decision which objects to stack. Overall, the learned symbols appeal to human intuition and reflect the task structure. For instance, a binary symbol with the meaning "next-to" which could be represented with RBFs was not learned since it does not help to predict and collect rewards here.

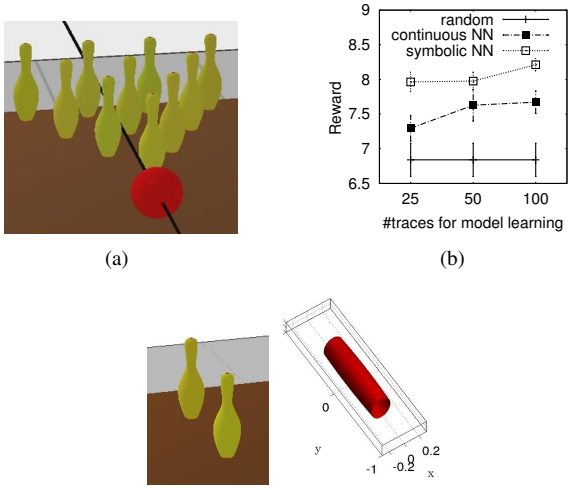
C. Simulated bowling scenario

For our second scenario we implemented a variant of bowling with the ODE physical simulator, trying to model as accurately as possible the setup of a ten-pin bowling game. There are 10 pins placed inside an equilateral triangle with a side length of 60cm. Each pin is described by a feature vector $y_i \in \mathbb{R}^6$ composed of the position dimensions



(a) (b) (c)

Fig. 3. *Simulated robot manipulation scenario.* (a) Simulator. (b) / (c) Two exemplary groundings of a relational symbol $b(o_1, o_2)$ learned in different runs. The red volumes denote the isosurface of the activation function of $g(x) = 0.5$ which is calculated from the continuous features $y_{1,2} = y_1 - y_2$, see Eq. (4); the axes are in meters. In (b), $b(o_1, o_2)$ captures that o_1 is directly on top of o_2 such as $b(o_{blue}, o_{orange})$. In (c), o_1 is above o_2 in the same stack such as $b(o_{green}, o_{blue})$.



(a) (b) (c)

Fig. 4. *Simulated bowling scenario.* (a) The goal is to knock down 10 simulated pins. The black line shows the ball path before deflections. (b) Results. (c) Exemplary learned binary symbol which indicates whether two objects have the same x-coordinate, i.e."in the same column" for a ball hit.

and the z-axis vector, see Figure 4(a). The action set was $\mathcal{A} = \{throwAtCenter(X), throwRightOfCenter(X)\}$: the first is a ball-throw straight (aligned with the y-axis) at the center of the target pin X, the second 7.5cm to the right of the center of X. This small translation has an effect on the ball trajectory, which is often deflected after collisions. The ODE simulation also has some noise in the collisions between ball and pins.

Our symbol learning method learned 1 unary and 3 binary symbols to describe the world, using training data $\mathcal{D}$ of 80 random ball throws in smaller worlds with 5 pins placed randomly on a subset of the 10 ball positions. We gathered traces of random throws with $n = 25, 50, 100$ and $T = 2$, i.e. the points are counted after 2 hits. We compared the performance of our methods in 100 bowling games with 2 hits each. Figure 4(b) indicates that the learned relational symbols capture essential information of the bowling world: SYMBOLIC-NN is the superior method. Figure 4(c) shows examples of a binary symbol we learned: it represents the basic geometric relation of two pins being on the same line w.r.t. the straight ball trajectories. Such a relation makes it likely that throwing the ball at one pin will also knock out the other. Another learned symbol can be interpreted as "behind and 15cm to the left", which is related to the possible ball deflections. The unary symbol could usually be interpreted as denoting "upright" pins.

D. Qualitative real-world experiment

We also performed also a qualitative study on a real robot platform, depicted in Fig. 1, which consists of a Schunk Light Weight arm with 7 DoF, a Schunk Dextrous Hand with 7 DoF, 6×14 tactile arrays on each of the 6 finger segments and a Bumblebee stereo camera. The robot is placed in front of a table with four different cans which it can recognize based on their color. The robot has motor primitives to grab and release cans. One unary and one binary relational symbol is learned from the experience of 30 random actions. Based on the learned symbols, the robot learns to plan efficiently sequences of motor primitives for building towers of cans. A video of this study can be found at `http://dl.dropbox.com/u/17220615/RSS-Submission/RSS-Extra-Material.html`. `http://dl.dropbox.com/u/17220615/RSS%20Submission/RSS%20Extra%20Material.html`.

VII. CONCLUSIONS AND FUTURE WORK

Symbolic representations are a promising approach to abstract reasoning of robots about sequences of motor primitives. But where do these symbols come from? We have proposed a bootstrapping approach to learn relational state symbols from geometric object features assuming the existence of a given set of motor primitives. We have demonstrated in an integrated system that our approach opens a door for real-world robots to leverage methods developed in AI for symbolic reinforcement learning and planning. These methods rely crucially on accurately grounded symbols.

Future work can combine our approach with active learning with a teacher where the robot generates situations in which it is uncertain about the symbolic grounding. Another major question is how to combine motor primitive learning methods with our symbol learning approach, aiming towards a mutual bootstrapping of motor primitives and respective symbolic representations.

APPENDIX

In several contexts we use a relational kernel $k(s, s')$ which measures the similarity of states. While we could use more elaborated existing relational kernels, here we resorted to a simple kernel that achieves generalization by exhaustively testing all permutations with respect to the identities of objects in the symbolic states. Let Π be the set of permutations of all M objects, containing $M!$ permutations $\pi \in \Pi$. Let $\kappa_\pi : \{0,1\}^\nu \to \{0,1\}^\nu$ denote the reordering s

according to the order of objects in π and thus of their predicate instantiations $p_k(o_m), p_l(o_m, o_{m'})$. We define a kernel k based on Π as $k(s, s') = \min_{\pi \in \Pi} \exp(\|s - \kappa_\pi(s')\|^2)$. This is the distance between s and the most similar permuted variation of s'. A similar relational kernel can also be applied on continuous features $y, y' \in \mathbb{R}^q$ composed of object features y_i. Let $\kappa_\pi : \mathbb{R}^q \to \mathbb{R}^q$ denote the reordering of the features y_i according to the permutation π. We get the kernel $k(y, y') = \min_{\pi \in \Pi} \exp(-\|y - \kappa_\pi(y')\|^2)$, showing that relational methods are not limited to symbolic discrete representations.

ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG), grants TO 409/1-3 and TO 409/7-1.

REFERENCES

[1] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *ICRA*, 2011, pp. 1470–1477.

[2] S. Harnad, "The symbol grounding problem," *Physica D*, vol. 42, pp. 335–346, 1990.

[3] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward understanding natural language directions," in *HRI*, 2010, pp. 259–266.

[4] F. J. Kurfess, "Neural networks and structured knowledge: Rule extraction and applications," *Applied Intelligence*, vol. 12, pp. 7–13, January 2000.

[5] A. Cangelosi, A. Greco, and S. Harnad, *Symbol Grounding and the Symbolic Theft Hypothesis*. Springer, London, 2002.

[6] D. Billman and J. Knutson, "Unsupervised concept learning and value systematicity: A complex whole aids learning the parts," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 22, no. 2, pp. 458 – 475, 1996.

[7] V. Truppa, E. Piano Mortari, D. Garofoli, S. Privitera, and E. Visalberghi, "Same/different concept learning by capuchin monkeys in matching-to-sample tasks," *PLoS ONE*, vol. 6, no. 8, p. e23809, 08 2011.

[8] A. Newell and H. Simon, "Gps: A program that simulates human thought," in *Computers and Thought*, Feigenbaum and Feldman, Eds. McGraw-Hill, New York, 1963.

[9] H. Dreyfus, *What Computers Can't Do: The Limits of Artificial Intelligence*. New York, USA: MIT Press, 1972.

[10] S. Džeroski, L. de Raedt, and K. Driessens, "Relational reinforcement learning," *Machine Learning Journal*, vol. 43, pp. 7–52, 2001.

[11] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *Science*, vol. 331, no. 6022, pp. 1279–1285, 2011.

[12] J. Feldman, "Symbolic representations of probabilistic worlds," *Cognition*, vol. 123, pp. 61–83, 2012.

[13] D. Katz, Y. Pyuro, and O. Brock, "Learning to manipulate articulated objects in unstructured environments using a grounded relational representation," in *RSS*, 2008, pp. 254–261.

[14] K. F. MacDorman, "Grounding symbols through sensorimotor integration," *Journal of the Robotics Society of Japan*, vol. 17, no. 1, pp. 20–24, 1999.

[15] E. Chinellato, A. Morales, E. Cervera, and A. P. D. Pobil, "Symbol grounding through robotic manipulation in cognitive systems," *Robotics and Autonomous Systems*, vol. 55, no. 12, pp. 851–859, 2007.

[16] H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling, "Learning symbolic models of stochastic domains," *Journal of Artificial Intelligence Research (JAIR)*, vol. 29, pp. 309–352, 2007.

[17] T. Lang and M. Toussaint, "Planning with noisy probabilistic relational rules," *Journal of Artificial Intelligence Research (JAIR)*, vol. 39, pp. 1–49, 2010.

[18] R. Parr, L. Li, G. Taylor, C. Painter-Wakefield, and M. L. Littman, "An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning," in *Proceedings of the 25th Int. Conf. on Machine learning*, ser. ICML 08, 2008, pp. 752–759.