Skill Learning by Autonomous Robotic Playing using Active Learning and Creativity

Simon Hangl, Vedran Dunjko, Hans J. Briegel and Justus Piater

Abstract—**We treat the problem of autonomous acquisition of manipulation skills where problem-solving strategies are initially available only for a narrow range of situations. We propose to extend the range of solvable situations by autonomous playing with the object. By applying previously-trained skills and behaviours, the robot learns how to prepare situations for which a successful strategy is already known. The information gathered during autonomous play is additionally used to learn an environment model. This model is exploited for active learning and the creative generation of novel preparatory behaviours. We apply our approach on a wide range of different manipulation tasks, e.g. book grasping, grasping of objects of different sizes by selecting different grasping strategies, placement on shelves, and tower disassembly. We show that the creative behaviour generation mechanism enables the robot to solve previously-unsolvable tasks, e.g. tower disassembly. We use success statistics gained during real-world experiments to simulate the convergence behaviour of our system. Experiments show that active improves the learning speed by around 9 percent in the book grasping scenario.**

Index Terms—**Active Learning, Hierarchical models, Skill Learning, Reinforcement learning, Autonomous robotics, Robotic manipulation, Robotic creativity**

I. INTRODUCTION

HUMANS perform complex object manipulations so effortlessly that at first sight it is hard to believe that this problem is still unsolved in modern robotics. This becomes less surprising if one considers how many different abilities are involved in human object manipulation. These abilities span from control (e.g. moving arms and fingers, balancing the body), via perception (e.g. vision, haptic feedback) to planning of complex tasks. Most of these are not yet solved in research by themselves, not to speak of combining them in order to design systems that can stand up to a comparison with humans. However, there is research on efficiently solving specific problems (or specific classes of problems) [1]–[5].

Not only the performance of humans is outstanding – most manipulation skills are learned with a high degree of autonomy. Humans are able to use experience and apply the previously learnt lessons to new manipulation problems. In order to take a step towards human-like robots we introduce a novel approach for autonomous learning that makes it easy to embed state-of-the-art research on specific manipulation problems. Further we aim to combine these methods in a unified framework which autonomously learns how to combine those methods and to solve increasingly complex tasks.

In this work we are inspired by the behaviour of infants at an age between 8 to 12 months. Piaget identified different phases of infant development [6]. A phase of special interest is the *coordination of secondary schemata* which he identifies as the stage of “first actually intelligent behaviour”. At this stage infants combine skills that were learned earlier in order to achieve more complex tasks, e.g. kicking an obstacle out of the way such that an object can be grasped. Children do not predict the outcome of actions and check the corresponding pre- and post conditions as it is done in many planning systems [7]–[9]. To them it is only important to know that a certain combination of manipulations is sufficient to achieve a desired task. The environment is prepared such that the actual skill can be applied without a great need for generalisation. Even adults exhibit a similar behaviour, e.g. in sports. A golf or tennis player will always try to perform the swing from similar positions relative to the ball. She will position herself accordingly instead of generalizing the swing from the current position. This is equivalent to concatenating two behaviours, walking towards the ball and executing the swing.

In previous work we introduced an approach that is loosely inspired by this paradigm [10]. The robot holds a set of *sensing actions*, *preparatory behaviours* and *basic behaviours*, i.e. behaviours that solve a certain task in a narrow range of situations. It uses the sensing actions to determine the state of the environment. Depending on the state, a preparatory behaviour is used to bring the environment into a state in which the task can be fulfilled by simple replay of the basic behaviour. The robot does not need to learn how to generalise a basic behaviour to every possibly observable situation. Instead, the best combination of sensing actions and preparatory behaviours is learned by autonomous playing.

We phrase the playing as a reinforcement learning (RL) problem, in which each rollout consists of the execution of a sensing action, a preparatory behaviour and the desired basic behaviour. Each rollout is time consuming, but not necessarily useful. If the robot already knows well what to do in a specific situation, performing another rollout in this situation does not help to improve the policy. However, if another situation is more interesting, it can try to prepare it and continue the play, i.e. *active learning*. Our original approach is model-free, which makes it impossible to exhibit such a behaviour. In this paper we propose to learn a forward model of the environment which allows the robot to perform transitions from *boring* situations to *interesting* ones. Another issue is the strict sequence of phases: *sensing* $\rightarrow$ *preparation* $\rightarrow$ *basic behaviour*. In this

S. Hangl and J. Piater are with the Department of Computer Science, University of Innsbruck, Austria; V. Dunjko is with the Institute of Theoretical Physics, University of Innnsbruck, Austria and with the Max Planck Institute for Physics Munich, Germany; H.J. Briegel is with the Institute of Theoretical Physics, University of Innsbruck, Austria and the Department of Philosophy, University of Konstanz, Germany.

work we weaken this restriction by enabling the robot to creatively generate novel preparatory behaviours composed of other already known behaviours. The environment model is used to generate composite behaviours that are potentially useful instead of randomly combining behaviours.

We illustrate the previously described concepts with the example of book grasping. This task is hard to generalise but easy to solve with a simple basic behaviour in a specific situation. The robot cannot easily get its fingers underneath the book in order to grasp it. In a specific pose, the robot can squeeze the book between two hands, lifting it at the spine and finally slide its fingers below the slightly-lifted book. Different orientations of the book would require adaption of the trajectory. The robot would have to develop some understanding of the physical properties, e.g. that the pressure has to be applied on the spine and that the direction of the force vector has to point towards the supporting hand. Learning this degree of understanding from scratch is a very hard problem.

Instead, we propose to use preparatory behaviours, e.g. *rotating the book by* $0°$*,* $90°$*,* $180°$ *or* $270°$, in order to move it to the correct orientation ($\phi = 0°$) before the basic behaviour is executed. The choice of the preparatory behaviour depends on the book's orientation, e.g. $\phi \in \{0°, 90°, 180°, 270°\}$. The orientation can be estimated by sliding along the book's surface, but not by poking on top of the book. The robot plays with the object and tries different combinations of sensing actions and preparatory behaviours. It receives a reward after executing the basic behaviour and continues playing. After training, the book grasping skill can be used as preparatory behaviour for other skills in order to build hierarchies.

If the robot already knows well that it has to perform the behaviour *rotate* $90°$ if $\phi = 270°$ and is confronted with this situation again, it cannot learn anything any more, i.e. it is *bored*. It can try to prepare a more interesting state, e.g. $\phi = 90°$ by executing the behaviour *rotate* $180°$. Further, if only the behaviour *rotate* $90°$ is available, the robot cannot solve the situations $\phi \in \{90°, 180°\}$ by executing a single behaviour. However, it can use behaviour compositions in order to generate the behaviours *rotate* $180°$ and *rotate* $270°$.

II. Related Work

A. Skill chaining and hierarchical reinforcement learning

Sutton et al. introduced the *options* framework for skill learning in a RL setting [11]. Options are actions of arbitrary complexity, e.g. atomic actions or high-level actions such as grasping, modelled by semi-Markov decision processes (SMDP). They consist of an option policy, an initiation set indicating the states in which the policy can be executed, and a termination condition that defines the probability of the option terminating in a given state. Options are orchestrated by Markov decision processes (MDP), which can be used for planning to achieve a desired goal. This is related to our notion of behaviours, however, behaviours are defined in a loser way. Behaviours do not have an initiation set and an explicit termination condition. Behaviours are combined by grounding them on actual executions by playing instead of concatenating them based on planning. Konidaris and Barto embedded so called *skill chaining* into the options framework [12]. Similar to our work, options are used to bring the environment to a state in which follow-up options can be used to achieve the task. This is done by standand RL techniques such as Sarsa and Q-learning. The used options themselves are autonomously generated, however, as opposed to our method, the state space is pre-given and shared by all options. Instead of autonomously creating novel options, Konidaris et. al. extended this approach by deriving options from segmenting trajectories trained by demonstration [13]. On a more abstract level, Colin et al. [14] investigated creativity for problem-solving in artificial agents in the context of hierarchical reinforcement learning by emphasising parallels to psychology. They argue that hierarchical composition of behaviours allows an agent to handle large search spaces in order to exhibit creative behaviour.

B. Model-free and model-based reinforcement learning in robotics

Our work combines a model-free playing system and a model-based creative behaviour generation system based on the environment model. Work on switching between model-free and model-based controllers was proposed in many areas of robotics [15]–[21]. The selection of different controllers is typically done by measuring the uncertainty of the controller's predictions. Renaudo et al. proposed switching between so called model-based and model-free experts, where the model is learned over time. The switching is done randomly [18], or by either majority vote, rank vote, Boltzmann Multiplication or Boltzmann Addition [19]. Similar work has been done in a navigation task by Caluwaerts et al. [20], [21]. Their biologically inspired approach uses three different experts, namely a taxon expert (model-free), a planning expert (model-based), and an exploration expert, i.e. exploring by random actions. A so called *gating network* selects the best expert in a given situation. All these methods hand over the complete control either to a model-based or a model-free expert. In contrast, our method always leaves the control with the model-free playing system which makes the final decision on which behaviours should be executed. The model-based system, i.e. behaviour generation using the environment model, is used to add more behaviours for model-free playing. This way, the playing paradigm can still be maintained while enabling the robot to come up with more complex ideas in case the task cannot be solved by the model-free system alone.

Dezfouli and Balleine sequence actions and group successful sequences to so-called *habits* [22]. Roughly speaking, task solutions are generated by a dominant model-based RL system and are transformed to atomic habits if they were rewarded many times together. In contrast, the main driving component of our method is a model-free RL system which is augmented with behavioural sequences by a model-based system. This way, the robot can deal with problems without requiring an environment model while still being able to benefit from it.

C. Developmental robotics

Our method shares properties with approaches in developmental robotics. A common element is the concept of *lifelong* *learning*, in which the robot develops more and more complex skills by interacting with the environment autonomously. Wörgötter et. al. proposed the concept of structural bootstrapping [23] in which knowledge acquired in earlier stages of the robot's life is used to speed up future learning. Weng provides an general description of a *self-aware and self-affecting agent* (SASE) [24]. He describes an agent with *internal* and *external* sensors and actuators respectively. It is argued that autonomous developmental robots need to be SASE agents and concrete implementations are given, e.g. navigation or speech learning. Our concept of boredom is an example of a pardigm, in which the robot decides on how to procede based on *internal sensing*. In general, developmental robotics shares some key concepts with our method, e.g. lifelong learning, incremental development or internal sensing. For a detailed discussion we refer to a survey by Lungarella et al. [25].

D. Active learning in robotics

In active learning the agent can execute actions which have an impact on the generation of training data [26]. In the simplest case, the agent explores the percept-action space by random actions [27]. The two major active learning paradigms, i.e. query-based and exploration-based active learning, differ in the action selection mechanism. Query-based learning systems request samples, e.g. by asking a supervisor for it. Typically, the request is based on the agent's uncertainty [28]–[30]. Chao et al. adopt query-based active learning for *socially guided machine learning* in robotics [31]. Task models are trained by interaction with a human teacher, e.g. classifying symbols assigned to tangram compounds. The robot could prepare a desired sample by itself, i.e. arranging interesting tangram compounds and asking the teacher for the class label. In contrast to our method, this is not done in practice, but the robot describes the desired compound.

Exploration-based active learning paradigms, on the other hand, select actions in order to reach states with maximum uncertainty [32]–[35]. Salganicoff et al. [36] and Morales et al. [37] used active learning for grasping. It was used to learn a prediction model of how good certain grasp types will work in a given situation. All these works deal with how to select actions such that a model of the environment can be trained more effectively. In our approach the training of the environment model is not the major priority. It is a side product of the autonomous play and is used to speed up learning and creatively generate behaviours on top of the playing system.

Kroemer et al. [38] suggested a hybrid approach of active learning and reactive control for robotic grasping. Active learning is used to explore interesting poses using an *upper confidence bound* (UCB) [39] policy that maximises the *merit*, i.e. the sum of the expected reward mean and variance. The actual grasps are executed by a reactive controller based on *dynamic movement primitives* (DMPs) [40] using attractor fields to move the hand towards the object and detractor fields for obstacle avoidance. This approach is tailored to a grasping task, in which the autonomous identification of possible successful grasps is hard due to high-dimensional search spaces. In contrast, our approach is acting on a more abstract level in which the described grasping method can be used as one of the preparatory behaviours. A more detailed investigation of active learning is outside the scope of this paper and can be found in a survey by Settles [41]. Special credit shall be given to work on *intrinsic motivation* [42]–[47]. It is a flavour of active learning which is commonly applied in autonomous robotics. Instead of maximising the uncertainty, these methods try to optimise for intermediate uncertainty. The idea is to keep the explored situations simple enough to be able to learn something, but complex enough to observe novel properties. Schmidhuber provides a sophisticated summary of work on intrinsic motivation and embedds the idea into a general framework [48]. He states that many of these works optimise some sort of intrinsic reward, which is related to the improvement of the prediction performance of the model. This is closely related to our notion of boredom, in which the robot rejects the execution of skills in a well-known situation for the sake of using to time on improving the policy in other situations. He further argues that such a general framework can explain concepts like creativity and fun.

E. Planning

Many of the previously mentioned methods are concerned with training forward models, which in consequence are used for planning in order to achieve certain tasks. Ugur et al. proposed a system that first learns action effects from interaction with the objects and is trained to predict single-object cagetories from visual perception [49]. In a second stage, multi-object interaction effects are learned by using the single-object categories, e.g. two solid objects can be stacked on top of each other. Discrete effects and categories are transformed into a PDDL description. Symbolic planning is used to create complex manipulation plans, e.g. for creating high towers by stacking. Konidaris et al. suggest a method in which symbolic state representations are completely determined by the agent's environment and actions [50]. They define a symbol algebra on the states derived from executed actions that can be used for high-level planning in order to reach a desired goal. Konidaris et al. extend this set-based formulation to a probabilistic representation in order to deal with the uncertainty observed in real-world settings [51]. A similar idea is present in our model-free approach, where the selection of sensing actions and the semantics of the estimated states depends on the desired skill.

All these approaches provide a method to build a bridge from messy sensor data and actions to high-level planning systems for aritifial intelligence. In order to so, similar to our approach, abstract symbols are used. However, these systems require quite powerful machinery in order to provide the required definition of pre- and post conditions for planning. In our approach the robot learns a task policy directly, which is augmented by a simple planning-based method for creative behaviour generation.

III. Problem Statement

The goal is to increase the scope of situations in which a *skill* can be applied by exploiting *behaviours*. A behaviour

$b \in B$ maps the complete (and partially unknown) state of system $\mathbf{e} \in A \times E$ to another state $\mathbf{e}' \in A \times E$ with

$$b : A \times E \mapsto A \times E \tag{1}$$

The sets A, E denote the internal state of the robot and the external state of the environment (e.g. present objects) respectively. We aim for autonomous training of a goal-directed behaviour, i.e. a *skill*. This requires a notion of success, i.e. by a success predicate. We define a skill $\sigma = (b^\sigma, \text{Success}^\sigma)$ as a pair of a *basic behaviour* b^σ, i.e. a behaviour that solves the task in a narrow range situations, and a predicate

$$\text{Success}^\sigma \left(b^\sigma \left(\mathbf{e}\right)\right) = true \tag{2}$$

with $\mathbf{e} \in D^\sigma$. The non-empty set $D^\sigma \subseteq A \times E$ is the set of all states in which the skill can be applied successfully, i.e. all states in which the fixed success predicate holds. We call the set D^σ the *domain of applicability* of the skill σ. The goal is to *extend the domain of applicability* by finding behaviour compositions $b_l \circ \cdots \circ b_2 \circ b_1$ with the property

$$\text{Success}^\sigma \left(b_l \circ \cdots \circ b_2 \circ b_1 \circ b^\sigma \left(\mathbf{e}\right)\right) = true \tag{3}$$

with $b_i \in B$ and $\mathbf{e} \in D'^\sigma \subseteq A \times E$ such that $D'^\sigma \supsetneq D^\sigma$, i.e. the domain of applicability is *larger* than before. A behaviour composition $b_l \circ \cdots \circ b_2 \circ b_1 \circ b^\sigma$ is a behaviour itself and therefore can be used to extend the domain of applicability of other skills. This way, skills can become more and more complex over time by constructing skill hierarchies.

IV. Contribution

We extend an approach for skill learning by autonomous playing introduced by Hangl et al. [10]. It uses only one preparatory behaviour per state, i.e. allowing only behaviour compositions of length $l = 1$, c.f. equation 3. This limitation enables the robot to perform model-free exploration due to the reduced search space. Allowing behaviour compositions of length $l > 1$ causes the learning problem to be intractable, but would help to solve more complex tasks.

Approaches dealing with problems of this complexity have to strongly reduce the search space, e.g. by symbolic planning [49]–[51]. We do not follow a planning-based paradigm in the traditional sense. The playing-based exploration of actions remains the core component of the system. In order to allow behaviour compositions of length $l > 1$ while still keeping the advantage of a small search space, we introduce a separate model-based system which generates potentially useful behaviour compositions. A forward model of the environment is trained with information acquired during autonomuous play. The environment model is used to generate new behaviour compositions that might be worth to be tried out. The ultimate decision whether a behaviour composition is used, however, is still up to the playing-based system. This way, the advantages of model-free and model-based approaches can be combined:

A) Behaviour compositions of arbitrary length can be explored without having to deal with the combinatorial explosion of possible behaviour compositions.
B) No or only weak modelling of the environment is required because the playing-based approach alone is still stable and fully-functional.
C) Exploration beyond the modelled percept-action space can still be done, e.g. a book flipping action can be used to open a box [10].

Proposals for novel preparatory behaviours are considered proportional to their expected usefulness. This enables the robot to first consider more conservative plans and to explore more unorthodox ideas in later stages. We refer to this procedure as *creative generation of behaviour proposals*. We relate to a principal investigation of creative machines [52], in which robots use a memory to propose combinations of previous experiences in order to exhibit *new* behavioural patterns.

We further exploit the environment model for speeding up the learning process by *active learning*. The robot can be *bored* of certain situations and is not only asking for different situations but also prepares them by itself. Whether or not the robot is bored is part of the internal state $\mathbf{e}_A \in A$ of the robot, which is made explicit in equation 1.

We believe that a lifelong learning robot must go through different developmental stages of increasing complexity. Optimally, these stages are not hard-coded to the system but emerge automatically over the course of the robot's life. We extend our original system such that these additional mechamisms are exploited as soon as the robot is ready for it, i.e. the environment model is mature enough.

V. Preliminaries

For better understanding of the remainder of the paper, we introduce the concept of perceptual states. We further provide a brief description of the core reinforcement learning method used in this paper – projective simulation (PS) [53].

A. Perceptual states

Let $\mathbf{e} \in A \times E$ be the complete physical state of the environment. In practice, it is impossible to estimate $\mathbf{e}$. However, only a facet of $\mathbf{e}$ is required to successfully perform a task. We use haptic exploration in order to estimate the relevant fraction of $\mathbf{e}$. A predefined set of sensing actions S is used to gather information. For many tasks only one sensing action $s \in S$ is required to estimate the relevant information, e.g. the book's orientation can be determined by sliding along the surface. While the sensing action s is executed, a multi-dimensional sensor data time series $M = \{\mathbf{t}_\tau\}$ of duration T with $\tau \in [1, \ldots, T]$ is measured. This time series is not the result of a deterministic process but follows an unknown probability distribution $p\left(M \mid \mathbf{e}, s\right)$.

In general, every state $\mathbf{e} \in A \times E$ potentially requires a different action to achieve the task successfully, e.g. how to grasp an object depends on the object pose. However, in many manipulation problems, similar states require a similar or even the same action. In these cases the state space can be divided into discrete classes e, e.g. the four orientations of a book in the book grasping task. We call such a class a *perceptual state*, denoted $e \in E^s_\sigma$. Note that the perceptual state space E^s_σ is not to be confused with the state space of environment E. The probability $p\left(e \mid M, s, \sigma\right)$ of a perceptual state e to be present depends on the measured sensor data M, the sensing action s and the skill σ for which the sensing action s is used,

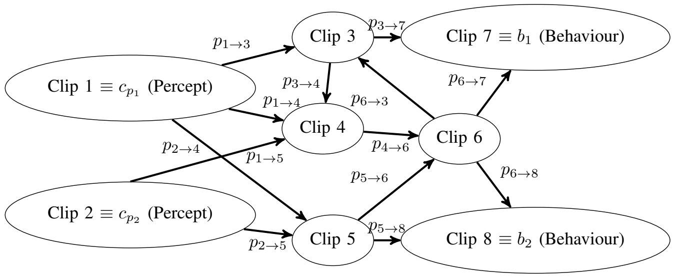


Fig. 1. Exemplary sketch of an episodic and compositional memory (ECM). A random walk always starts at a percept clip (e.g. clip 1, clip 2) and ends with an behaviour clip (e.g. clip 7, clip 8). A transition $c \to c'$ from clip c to clip c' is done with the probability $p_{c\to c'}$.

e.g. poking in book grasping means something different than in box opening. The perceptual state spaces of two sensing actions $s, s' \in S$ can coincide, partly overlap or be distinct e.g. sliding along the surface allows the robot to estimate the orientation of a book, whereas poking does not.

B. Projective simulation

Projective simulation (PS) [53] is a framework for the design of intelligent agents and can be used for reinforcement learning (RL). PS was shown to exhibit competitive performance in several reinforcement learning scenarions ranging from classical RL problems to adaptive quantum computation [54]–[57]. It is a core component of our method and was chosen due to structural advantages, conceptual simplicity and good extensibility. We briefly describe the basic concepts and the modifications applied in this paper. A detailed investigation of its properties can be found in [55].

Roughly speaking, the PS agent learns the probability distribution $p\left(b \,|\, \lambda,\, \mathbf{e}\right)$ of executing a behaviour b (e.g. a preparatory behaviour) given the observed sensor data λ (e.g. a verbal command regarding which skill to execute) in order to maximise a given reward function $r\left(b, \lambda,\, \mathbf{e}\right)$. In this paper, reward is given if $\text{Success}^{\sigma}\left(b \circ b^{\sigma}\left(\mathbf{e}\right)\right) = \text{true}$, given a command λ to execute skill σ in the present environment state $\mathbf{e}$. Note that the state $\mathbf{e}$ is never observed directly. Instead, perceptual states are estimated throughout the skill execution.

In general, the core of the PS agent is the so-called *episodic and compositional memory* (ECM). An exemplary sketch of an ECM is shown in Fig. 1. It stores fragments of experience, so-called *clips*, and connections between them. Each clip represents a previous experience, i.e. percepts and actions.

The distribution $p\left(b \,|\, \lambda,\, \mathbf{e}\right)$ is updated after a rollout, i.e. observing a percept, choosing and executing a behaviour according to $p\left(b \,|\, \lambda,\, \mathbf{e}\right)$, and receiving reward from the environment. The distribution $p\left(b \,|\, \lambda,\, \mathbf{e}\right)$ is implicitly specified by assigning transition probabilities $p_{c\to c'} = p\left(c' \,|\, c\right)$ to all pairs of clips $\left(c,\, c'\right)$ (in Fig. 1 only transitions with probability $p_{c\to c'} \neq 0$ are visualised). Given a certain *percept clip*, i.e. a clip without inbound transitions like clips 1 and 2, the executed *behaviour clip*, i.e. a clip without outbound transitions like clips 7 and 8, is selected by a *random walk* through the ECM. A random walk is done by hopping from clip to clip according to the respective transition probabilities until a behaviour is reached. Clips are discrete whereas sensor data is typically continuous, e.g. voice commands. A domain-specific input coupler distribution $I(c_p \,|\, \lambda, \mathbf{e})$ modelling the probability of observing a discrete percept clip c_p given an observed signal λ is required. The distribution $p\left(b \,|\, \lambda, \mathbf{e}\right)$ is given by a random walk through the ECM with

$$p\left(b \,|\, \lambda, \mathbf{e}\right) = \sum_{c_p} \left(I(c_p \,|\, \lambda, \mathbf{e}) \sum_{w \in \Lambda(a, c_p)} p(b \,|\, c_p, w) \right) \quad (4)$$

where $p(b \,|\, c_p, w)$ is the probability of reaching behaviour b from percept c_p via the path $w = (c_p = c_1, c_2, \ldots, c_K = b)$. The set $\Lambda(b, c_p)$ is the set of all paths from the percept clip c_p to the behaviour clip b. The path probability is given by

$$p(b \,|\, c_p, w) = \prod_{j=1}^{K-1} p(c_{j+1} \mid c_j) \quad (5)$$

The agent learns by adapting the probabilities $p_{c\to c'}$ according to the received reward (or punishment) $r \in \mathbb{R}$. The transition probability $p_{c\to c'}$ from a clip c to another clip c' is specified by the abstract transition weights $h \in \mathbb{R}^{+}$ with

$$p_{c\to c'} = p\left(c \mid c'\right) = \frac{h_{c\to c'}}{\sum_{\hat{c}} h_{c\to \hat{c}}} \quad (6)$$

After each rollout, all weights $h_{c\to c'}$ are updated. Let w be a random walk path with reward $r^{(t)} \in \mathbb{R}$ at time t. The transition weights are updated according to

$$h^{t+1}_{c\to c'} = \max\left(1,\, h^{t}_{c\to c'} - \zeta\left(h^{t}_{c\to c'} - 1\right) + \rho\left(c, c', w\right) r^{(t)}\right) \quad (7)$$

where $\rho(c, c', w)$ is 1 if the path w contains the transition $c \to c'$ and 0 otherwise. The *forgetting factor* ζ defines the rate with which the agent forgets previously learned policies.

VI. Skill learning by robotic playing

The following section describes the method for autonomous skill acquisition by autonomous playing on which this work is based on [10]. The sections VII – IX present extensions that run in parallel and augment the autonomous playing.

A. ECM for robotic playing

A skill σ is executed by a random walk through the layered ECM shown in Fig. 2. It consists of the following layers:

A) Input couplers: Input couplers map user commands about which skill to execute to the corresponding skill clip. The percept of this ECM is not the state of the environment, but the command of which skill to execute.
B) Desired skills: Each clip σ, i.e. a percept clip, represents a skill the robot is able to perform.
C) Sensing actions: Each clip $s \in S$ corresponds to one sensing action. All skills share the same sensing actions.
D) Perceptual states: Each clip $e \in E^{s}_{\sigma}$ corresponds to a perceptual state under the sensing action s for the skill σ. Note that the perceptual states are different for each skill-sensing action pair (σ, s) and typically do not have the same semantics, e.g. the states under sensing action

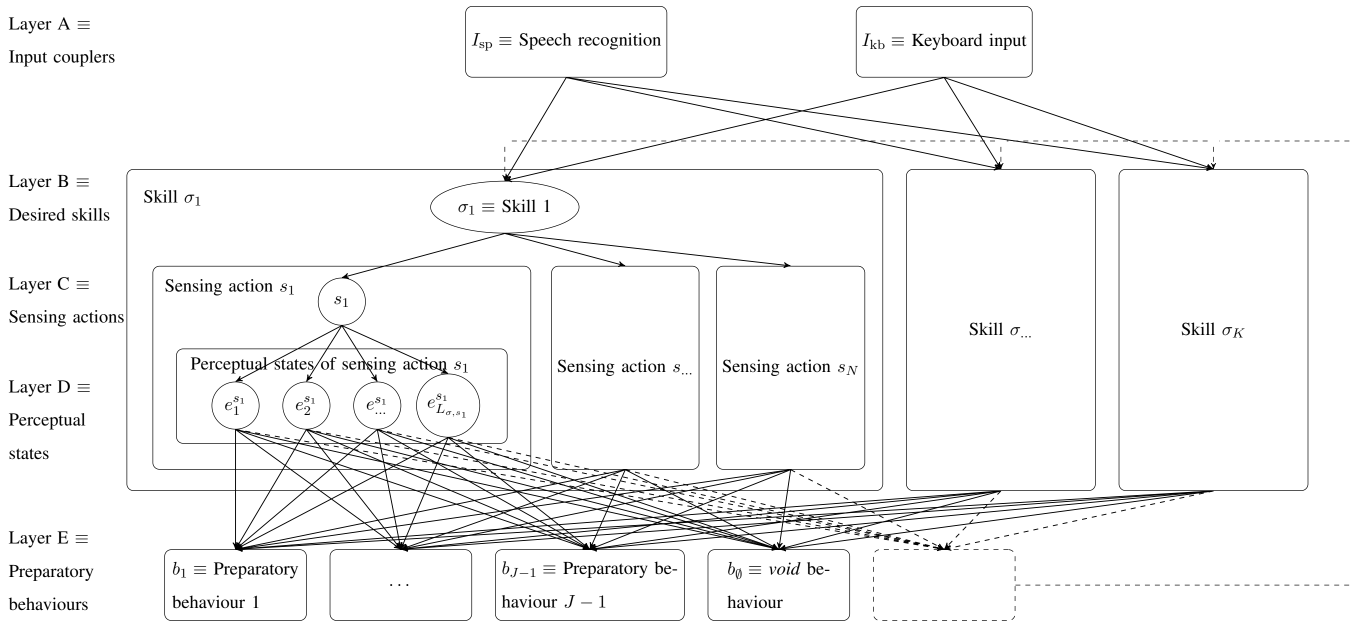


Fig. 2. ECM for autonomous robotic playing. For execution a random walk is performed from layer A to layer E. The transition from layer C to layer D is performed by executing the corresponding sensing action s, measuring the haptic data and using a time series classifier. All other transitions follow equation 6. The preparatory behaviour $b_\emptyset \equiv$ (*void* behaviour) is always in the set of preparatory behaviours. The dashed box and lines refer to skills used as preparatory behaviours in order to build skill hierarchies. After preparation, the *basic* behaviour b^σ corresponding to the desired skill σ is executed.

$s \in S$ might identify the object pose, whereas the states under $s' \in S$ might denote the object's concavity.

E) Preparatory behaviours: Each clip corresponds to a behaviour which can be atomic (solid transitions) or other trained skills (dashed transitions). Since the basic behaviour b^σ of a skill was shown to the robot in one perceptual state, there is at least one state that does not require preparation. Therefore, the *void*-behaviour $b_\emptyset$, in which no preparation is done, is in the set of behaviours.

The robot holds the sets of skills $\{\sigma = (b^\sigma, \mathrm{Success}^\sigma)\}$, sensing actions S (e.g. sliding, poking, pressing) and preparatory behaviours B (e.g. pushing). A skill is executed by performing a random walk through the ECM and by performing the actions along the path. The idle robot waits for a skill execution command λ whic is mapped to skill clips in Layer B by coupler functions, e.g. I_{kb} and I_{sp} mapping a keyboard input / voice commands to the desired skill clip σ. A sensing action $s \in S$ is chosen and executed according to the transition probabilities and a sensor data time series M is measured. The perceptual state $e \in E_\sigma^s$ is estimated from M. This transition is done deterministically by a classifier and not random as in the steps before. Given the perceptual state e, the environment is prepared by executing a behaviour $b \in B$. Finally, the basic behaviour b^σ is executed. If a basal bevahiour of a skill requires an object to be grasped, only the sensing action *weighing* is available in order to estimate whether an object is grasped. We stress that this is only a restriction enforced due to practical considerations and is not required in principle.

B. Skill Training

A novel skill $\sigma = (b^\sigma, \mathrm{Success}^\sigma)$ is trained by providing the basic behaviour b^σ for a narrow range of situations, e.g. by hard coding or learning from demonstration [4], [13], [58]–[62]. The domain of applicability is extended by learning:

a) which sensing action should be used to estimate the relevant perceptual state;
b) how to estimate the perceptual state from haptic data;
c) which preparatory behaviour helps to achieve the task in a given perceptual state.

The skill ECM (Fig. 2) is initialised in a meaningful way (sections VI-B1, VI-B2) and afterwards refined by executing the skills and collecting rewards, i.e. *autonomous playing*.

1) Haptic database creation: In a first step, the robot creates a haptic database by exploring how different perceptual states "feel", c.f. problem b). It performs all sensing actions $s \in S$ several times in all perceptual states e^s, acquires the sensor data M and stores the sets $\{(e^s, s, \{M\})\}$. With this database the distribution $p(e \mid M, s, \sigma)$ (section V-A) can be approximated and a perceptual state classifier is trained.

There are two ways of preparing different perceptual states. Either the supervisor prepares the different states (e.g. all four book poses) or the robot is provided with information on how to prepare them autonomously (e.g. *rotate by 90°* produces all poses). In the latter case the robot assumes that after execution of the behaviour a new perceptual state e' is present and adds it to the haptic database. This illustrates three important assumptions: The state $e^s \in E_\sigma^s$ is invariant under the sensing action $s \in S$ (e.g. the book's orientation remains the same irrespective of how often *sliding* is executed) but not under preparatory behaviours $b \in B$ (e.g. the book's orientation changes by using the *rotate* 90° behaviour), which yields

$$e^s \xrightarrow{s} e^s \tag{8}$$

$$e^s \xrightarrow{b} e'^s \tag{9}$$

Further we do not assume that a sensing action s' leaves the perceptual state e^s of another sensing action s unchanged (e.g. sliding softly along a tower made of cups does not change the position of the cups whereas poking from the side may cause the tower to collapse). This insight is reflected by the example

$$e^s \xrightarrow{s} e^s \xrightarrow{s} \ldots \xrightarrow{s} e^s \xrightarrow{s'} e^{s'} \xrightarrow{s} e'^s \qquad (10)$$

2) ECM Initialisation: The ECM in Fig. 2 is initialised with the uniform transition weights h_{init} except for the weights between layers B and C. These weights are initialised such that the agent prefers sensing actions $s \in S$ that can discriminate well between their environment states $e^s \in E^s_\sigma$. After the generation of the haptic database the robot performs cross-validation for the perceptual state classifier of each sensing action $s \in S$ and computes the average success rate r_s. A *discrimination score* D_s is computed by

$$D_s = \exp\left(\alpha r_s\right) \qquad (11)$$

with the free parameter α called *stretching factor*. The higher the discrimination score, the better the sensing action can classify the corresponding perceptual states. Therefore, sensing actions with a high discrimination score should be preferred over sensing actions with a lower score. The transition weights between all pairs of the skill clip σ and the sensing action clips $s \in S$ are initialised with $h_{\sigma \to s} = D_s$. We use a C-SVM classifier implemented in LibSVM [63] for state estimation.

3) Extending the domain of applicability: The domain of applicability of a skill σ is extended by running the PS as described in section V-B on the ECM in Fig. 2. The robot collects reward after each rollout and updates the transition probabilities accordingly. Skills are added as preparatory behaviours of other skills as soon as they are *well-trained*, i.e. the average reward $\bar{r}$ over the last t_{thresh} rollouts reaches a threshold $\bar{r} \geq r_{\text{thresh}}$. This enables the robot to create increasingly complex skill hierarchies. The complete training procedure of a skill σ is shown in Fig. 3. Only the non-shaded parts and solid transitions are available in this basic version.

C. Properties and extensions

A strong advantage is that state-of-the-art research on object manipulation can be embedded by adding the controllers to the set of behaviours. Algorithms for specific problems (e.g. grasping, pushing [64]–[68]) can be re-used in a bigger framework that orchestrates their interaction.

In the basic version the state space is comparatively small, which enables the robot to learn skills without an environment model. Further, the robot to learn fast while still preserving the ability to learn quite complex skills autonomously. However, the lack an environment model can be both an advantage and a disadvantage. Testing a hypothesis directly on the environment enables the robot to apply behaviours outside of the intended context (e.g. a book flipping behaviour might be used to open a box [10]). This is hard to achieve with model-based approaches if the modeled domain of a behaviour cannot properly represent the relevant information. On the other hand, the lack of reasoning abilities limits the learning speed and the complexity of solvable problems. We overcome

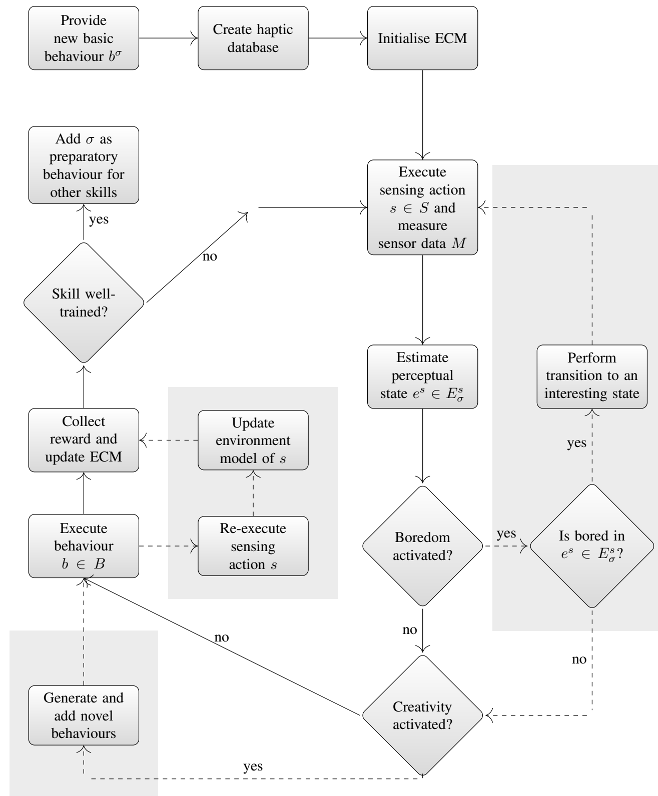


Fig. 3. Flow chart of the skill training procedure. A novel skill is trained by showing a new basic behaviour. The robot extends the domain of applicability by playing the object, i.e. by performing a random walk through the network shown in Fig. 2. The solid lines indicate the behaviour of the basic approach [10]. The shaded areas and dashed lines show the proposed extensions, i.e. *training of an environment model*, *boredom* and *creative skill generation*.

this problem by additionally learning an environment model from information acquired during playing. The robot learns a distribution of the effects of behaviours on given perceptual states by re-estimating the state after execution. We use the environment model for two purposes: *active learning* and *creative generation of novel preparatory behaviours*.

The basic version intrinsically assumes that all required preparatory behaviours are available. This constitutes a strong prior and limits the degree of autnomoy. We weaken this requirement by allowing the robot to creatively generate potentially useful combinations of behaviours. These are made avaible for the playing system which tries them out. Further, experiments showed that the learning speed was decreased by performing rollouts in situations that were already solved before. We use the environment model to implement *active learning*. Instead of asking a supervisor to prepare interesting situations, the robot prepares them by itself.

VII. Learning an Environment Model

The *environment model* predicts the effect, i.e. the resulting perceptual state, of a behaviour on a given perceptual state. An environment model is the probability distribution $p\left(e'^s \mid e^s, b, \sigma\right)$ where $e^s, e'^s \in E^s_\sigma$ are perceptual states of the sensing action $s \in S$ for a skill σ, and $b \in B$ is a behaviour. It denotes the probablity of the transition $e^s \xrightarrow{b} e'^s$. The required

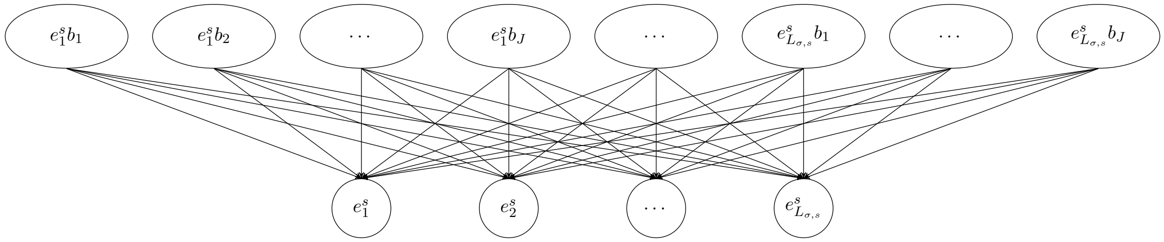


Fig. 4. ECM for the environment model under the sensing action s. It reflects the probability of state transitions $e^s \xrightarrow{b} e'^s$. The environment model is trained by adding a sensing step after executing the preparatory behaviour $b \in B$.

training data is acquired by re-executing the sensing action s after applying the behaviour b, c.f. shaded center part in Fig. 3. Given a playing sequence $\sigma \xrightarrow{s} e^s \xrightarrow{b} e'^s$ (c.f. Fig. 2) the effect can be observed by re-executing s with

$$\sigma \xrightarrow{s} e^s \xrightarrow{b} e'^s \xrightarrow{s} e'^s \tag{12}$$

The assumptions in equations 8 - 9 forbid to additionally execute other sensing actions $s' \in S$ without influencing the playing based method. This limitation prevents the robot from learning more complex environment models as done in related work [42]–[47], e.g. capturing transitions between perceptual states of different sensing actions. However, the purpose of the environment model is not to perform precise plans but to feed the core playing component with new *ideas*.

We represent the distribution $p\left(e'^s \,|\, e^s,\, b,\, \sigma\right)$ by another ECM for each skill - sensing action pair (σ, s) as shown in Fig. 4. The percept clips consist of pairs (e^s, b) of perceptual states $e^s \in E_\sigma^s$ and preparatory behaviours $b \in B$. The target clips are the possible resulting states $e'^s \in E_\sigma^s$. The environment model is initialised with uniform weights $h^{\text{env}}_{(e^s,b)\rightarrow e'^s} = 1$. When a skill σ is executed using the path in equation 12, a reward of $r^{\text{env}} \in \mathbb{R}^+$ is given for the transition

$$(e^s, b) \rightarrow e'^s \tag{13}$$

and the weights are updated accordingly, c.f. equation 7. When a novel preparatory behaviour b_{K+1} is available for playing, e.g. a skill is well-trained and is added as a preparatory behaviour, it is included into the environment models for each skill - sensing action pair (σ, s) by adding clips (e^s, b_{K+1}) for all states $e^s \in E_\sigma^s$ and by connecting them to all $e'^s \in E_\sigma^s$ with the uniform initial weight $h^{\text{env}}_{\text{init}} = 1$.

We employ a practical restriction on the scope of the environment model. The additional sensing action execution is only done if the grasp outcome of the seleced preparatory behaviour and the grasp requirement of the the sensing action match, e.g. if the preparatory behaviour grasps the object, but the sensing action was *sliding*, re-execution of the sensing action would destroy the grasp and is not done.

VIII. Autonomous Active Learning

In the basic version an optimal selection of observed perceptual states is required in order to learn the correct behaviour in all possible states, i.e. in a semi-supervised setting a human supervisor should mainly prepare unsolved perceptual states. This would require the supervisor to have knowledge about the method itself and about the semantics of perceptual states, which is an undesirable property. Instead, we propose to equip the robot with the ability to reject perceptual states in which the skill is well-trained already. In an autonomous setting, this is not sufficient as it would just stall the playing. The robot has to prepare a more interesting state autonomously. We propose to plan state transitions by using the environment model in order to reach states which (i) are *interesting* and (ii) can be prepared with high confidence. We can draw a loose connection to human behaviour. In that spirit, we call the rejection of well-known states *boredom*.

A. Boredom

The robot may be *bored* in a given perceptual state, if it is *confident* about the task solution, i.e. if the distribution of which preparatory behaviour to select is highly concentrated. In general, every function reflecting uncertainty can be used. We use the normalised Shannon entropy to measure the confidence in a perceptual state $e \in E_\sigma^s$, given by

$$\hat{H}_e = \frac{H\left(b \,|\, e\right)}{H_{\max}} = -\frac{\sum_{b' \in B} p\left(b = b' \,|\, e\right) \log_2 p\left(b = b' \,|\, e\right)}{\log_2 J} \tag{14}$$

where J is the number of preparatory behaviours. If the entropy is high, the robot either has not learned anything yet (and therefore all the transition weights are close to uniform) or it observes the degenerate case that all preparatory behaviours deliver (un)successful execution (in which case there is nothing to learn at all). If the entropy is low, few transitions are strong, i.e. the robot knows well how to handle this situation. We use the normalised entropy to define the probability of being bored in a state $e \in E_\sigma^s$ with

$$p\left(\text{bored} = true \,|\, e\right) = 1 - \beta \hat{H}_e \tag{15}$$

The constant $\beta \in [0, 1]$ defines how *immune* the agent is to boredom. The robot samples according to $p\left(\text{bored} \,|\, e\right)$ and decides on whether to refuse the execution.

B. Transition Confidence

If the robot is bored in a perceptual state $e' \in E_\sigma^s$, it autonomously tries to prepare a more interesting state $\hat{e} \in E_\sigma^s$. This requires the notion of a *transition confidence* for which the environment model can be used. We aim to select behaviours conservatively which allows the robot to be certain about the effect of the transition. We do not use the probability of reaching one state from another directly, but use a measure considering the complete distribution $p(e \mid e', b)$. By maximising the normalised Shannon entropy, we favour deterministic transitions. For each state-action pair (e', b) in Fig. 4 we define the transition confidence $\nu^s_{e'b}$ by

$$\nu^s_{e'b} = 1 - \frac{H\left(e \mid (e', b)\right)}{H_{\max}} = 1 - \frac{H\left(e \mid (e', b)\right)}{\log_2 L_{\sigma,s}} \tag{16}$$

where $e' \in E^s_\sigma$, $b \in B$, and $L_{\sigma,s}$ is the number of perceptual states under the sensing action $s \in S$, i. e. the number of children of the clip (e', b). In contrast to the entropy computed in section VIII-A, the transition confidence is computed on the environment model, c.f. Fig. 4. The *successor function* $\mathrm{su}(e, b)$ returns the most likely resulting outcome of executing behaviour b in a perceptual state $e \in E^s_\sigma$ and is defined by

$$\mathrm{su}(e, b) = \arg\max_{e'} p_{(e,b) \to e'} \tag{17}$$

In practice, single state transitions are not sufficient. For paths $e = e^s_1 \xrightarrow{b_1} e^s_2 = \mathrm{su}(e^s_1, b_1) \xrightarrow{b_2} \ldots \xrightarrow{b_{L-1}} \mathrm{su}(e^s_{n_{L-1}}, b_{L-1}) = e^s_L = e'$ of length L we define the transition confidence with

$$\nu^s_{e\mathbf{b}} = \prod_{l=1}^{L-1} \nu^s_{e_l b_l} \tag{18}$$

where the vector $\mathbf{b} = (b_1, b_2, \ldots, b_{L-1})$ denotes the sequence of behaviours. This is equivalent to a greedy policy, which provides a more conservative estimate of the transition confidence and eliminates consideration of transitions that could occur by pure chance. A positive side effect is the efficient computation of equation 18. Only the confidence of the most likely path is computed instead of iterating over all possible paths. The path $\mathbf{b}$ is a behaviour itself and the successor is given by $\mathrm{su}(e, \mathbf{b}) = \mathrm{su}(e^s_{n_{L-1}}, b_{L-1})$.

C. Active Learning

If the robot encounters a boring state $e \in E^s_\sigma$, the goal is to prepare the most interesting state that can acutally be produced. We maximise the *desirability function* given by

$$(\mathbf{b}, L) = \arg\max_{\mathbf{b}, L} \left[\hat{H}_{\mathrm{su}(e,\mathbf{b})} \nu_{e\mathbf{b}} + \frac{\epsilon}{\mathrm{cost}\left(\mathbf{b}\right)}\right] \tag{19}$$

where $\hat{H}_{\mathrm{su}(e,\mathbf{b})}$ is the entropy of the expected final state and $\nu_{e\mathbf{b}}$ is the confidence of reaching the final state by the path $\mathbf{b}$. The *balancing factor* ϵ defines the relative importance of the desirability and the path cost. The path cost $\mathrm{cost}\left(\mathbf{b}\right)$ can be defined by the length of the path L, i.e. penalising long paths, or, for instance, by the average execution time of $\mathbf{b}$. Equation 19 balances between searching for an interesting state while making sure that it is reachable. In practice it can be optimised by enumerating all paths of reasonable length, e.g. $L < L_{\max}$, with typical values of $L_{\max} \leq 4$.

The basic method is extended by sampling from the boredom distribution after the state estimation. If the robot is bored, it optimises the desirability function and executes the transition to a more interesting state. This is followed by restarting the skill execution with boredom turned off in order to avoid boredom loops, c.f. right shaded box in Fig. 3.

IX. Creative Behaviour Generation

In many cases, the required preparatory behaviour is a combination of other available behaviours, e.g. *rotate* 180° ≡ *rotate* 90° + *rotate* 90°. Without using some sort of intelligent reasoning, the space of concatenated behaviours explodes and becomes intractable. However, any sequence of behaviours that transfers the current unsolved state to a target state, i.e. a state which does not require any preparation, is potentially useful as a compound behaviour itself. Sequences can be generated by planning transitions to target states. If the robot is bored, it uses active learning, if not, the situation is not solved yet and novel compound behaviours might be useful.

A perceptual state $e_\emptyset \in E^s_\sigma$ is a target state if the transition with the highest probability in the playing ECM (Fig. 2) leads to the *void*-behaviour with $p_{e_\emptyset \to b_\emptyset}$. If there exists a path $e^s \xrightarrow{\mathbf{b}} e_\emptyset$ from the current perceptual state $e^s \in E^s_\sigma$ to a target state $e_\emptyset$, the sequence $\mathbf{b} = (b_1, \ldots, b_L)$ is a candidate for a novel behaviour. The robot is *curious* about trying out the novel compound behaviour $\mathbf{b}$, if the transition confidence $\nu_{e^s\mathbf{b}}$, with $\mathrm{su}(e^s, \mathbf{b}) = e_\emptyset$, and the probability $p_{e_\emptyset \to b_\emptyset}$ of the state actually being a real target state are both high. This is measured by the *curiosity score* of the compound behaviour given by

$$\mathrm{cu}(e^s, \mathbf{b}) = \nu_{e^s\mathbf{b}}\, p_{\mathrm{su}(e^s,\mathbf{b}) \to b_\emptyset} \tag{20}$$

The factor $p_{\mathrm{su}(e^s,\mathbf{b}) \to b_\emptyset}$ reduces the score in case the state $e_\emptyset$ is a target state with low probability. This can happen if in previous rollouts all other behaviours were executed and were punished. We use a probability instead of a confidence value to allow creativity even in early stages where a target state was not identified with a high probability.

The compound behaviour with the highest score is added as novel behaviour $b_{J+1} = \mathbf{b}$ with the probability given by squashing the curiosity score into the interval $[0, 1]$ with

$$p\left(\text{add } b_{J+1} = \mathbf{b} \mid e^s\right) = \mathrm{sig}\left[\gamma\, \mathrm{cu}\left(e^s, \mathbf{b}\right) + \delta\right] \tag{21}$$

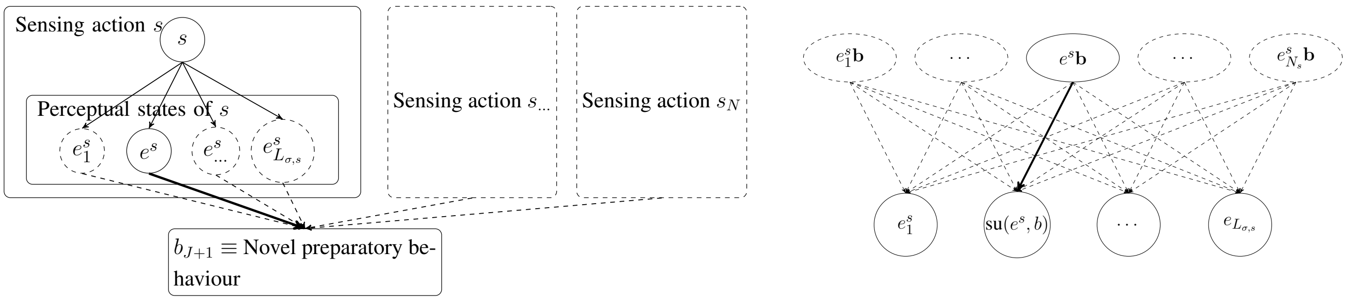
where sig is the logistic sigmoid. The parameters γ, δ define how conservatively novel behaviour proposals are created. The novel behaviour b_{J+1} is added as preparatory behaviour for all perceptual states under the current skill σ with the weights

$$h_{e \to b_{J+1}} = \begin{cases} h_{\mathrm{init}}\left[1 + \mathrm{cu}\left(e^s, b_{J+1}\right)\right] & , \text{ if } e = e^s \\ h_{\mathrm{init}} & , \text{ else} \end{cases} \tag{22}$$

It is added with at least the initial weight h_{init}, but increased proportional to the curiosity score for the current perceptual state $e^s \in E^s_\sigma$, c.f. Fig. 5a. The novel behaviour is also inserted to the environment model of all sensing actions $s \in S$. For each perceptual state $e \in E^s_\sigma$, a clip (e, b) is added and connected to the clips $e' \in E^s_\sigma$ in second layer with the weights

$$h_{(e,b) \to e'} = \begin{cases} h_{\min}\left(b\right) & , \text{ if } e = e^s,\ e' = \mathrm{su}(e^s, b),\ b = b_{J+1} \\ h^{\mathrm{env}}_{\mathrm{init}} & , \text{ else} \end{cases} \tag{23}$$

where $h_{\min}\left(b_{J+1}\right) = h_{\min}\left(\mathbf{b}\right)$ is the minimum transition value on the path $\mathbf{b}$ through the environment model, following the idea that a chain is only as strong as its weakest link. The weights of all other transitions are set to the initial weight $h^{\mathrm{env}}_{\mathrm{init}}$, c.f. Fig. 5b.



(a) The behaviour b_{J+1} is added the initial weight h_{init} (dashed lines) to all perceptual states except for the current state e^s. In this case it is added with a higher weight proportional to the curiosity score (solid line), c.f. equation 22. The connections to all other preparatory behaviours are ommitted in this figure.

(b) All pairs (e, b_{J+1}) of perceptual states $e \in E_\sigma^s$ and the behaviour $b_{J+1} = \mathbf{b}$ are added. The weights are chosen according to equation 23 (case 1: solid line, case 2: dashed lines). Note that case 1 only applies for the currently used sending action $s \in S$.

Fig. 5. Insertion of the novel compound behaviour $b_{J+1} = \mathbf{b}$, creatively generated in the current perceptual state $e^s \in E_\sigma^s$, to the playing ECM of skill σ (left), c.f. Fig. 2, and the environment models of the used sensing action $s \in S$ (right) respectively.

X. Experiments

We evaluate our method using a mix of simulated and real-world experiments. Our real-world experiments cover a wide range of skills to show the expressive power. We show how skill hierarchies are created within our framework. Success statistics of the single components (sensing accuracy, success rate of preparatory behaviours, success rate of basic behaviours) were used to assess the convergence behaviour by simulation. Table I lists the used parameter values. We execute all skills and behaviours in impedance mode in order to prevent damage to the robot. Further, executed behaviours are stopped if a maximum force is exceeded. This is a key aspect for model-free playing, which enables the robot to try out arbitrary behaviours in arbitrary tasks.

Parameter	Name	Values
Skill success reward	$r(\text{success})$	1000
Skill failure punishment	$r(\text{failure})$	-30
PS forgetting factor	ζ	0
Environment model reward	r^{env}	10
Skill ECM intial weight	h_{init}	200
Environment model ECM intial weight	$h_{\text{init}}^{\text{env}}$	1
Stretching factor	α	25
Boredom immunity	β	0.8
Squashing scale	γ	0.1
Squashing shift	δ	0.95
Balancing factor	ϵ	0.1
Maximum creativity path length	$L_{\max}$	4

TABLE I
List of free parameters and values used

A. Experimental Setup

The robot setting is shown in Fig. 6. For object detection a Kinect mounted above the robot is used. All required components and behaviours are implemented with the *kukadu* robotics framework[1]. The perceptual states are estimated from joint positions, Cartesian end-effector positions, joint forces and Cartesian end-effector forces / torques. Objects are localised by removing the table surface from the point cloud and fitting a box by using PCL. Four controllers implement the available preparatory behaviours:

- *Void behaviour*: The robot does nothing.
- *Rotation*: The object is rotated by a circular finger movement around the object's center. The controller can be parametrised with the approximate rotation angle.
- *Flip*: The object is squeezed between the hands and one hand performs a circle with the radius of the object in the XZ-plane which yields a vertical rotation.
- *Simple grasping*: The gripper is positioned on top of the object and the fingers are closed.

The haptic database consists of at least 10 samples per perceptual state. Before sensing, the object is pushed back to a position in front of the robot. We use four sensing actions:

- *No Sensing*: Some tasks do not require any prior sensing and have only one state. The discrimination score is computed with a success rate of $r_s = 0.5$, c.f. equation 11.
- *Slide*: A finger is placed in front of the object. The object is pushed towards the finger with the second hand until contact or until the hands get too close to each other (safety reasons). Sensig is done by bending the finger.
- *Press*: The object is pushed with one hand towards the second hand until the force exceeds a certain threshold.
- *Poke*: The object is poked from the top with a finger.
- *Weigh*: Checks a successful grasp by measuring the z-component of the Cartesian force.. The perceptual states are fixed, i.e. not grasped / grasped.

B. Real-world tasks

We demonstrate the generality of our method in several scenarios. Each skill can use the described preparatory behaviours, and additionally, the skills trained before. If not stated otherwise, all basic behaviours are dynamic movement primitives (DMPs) [40] trained by kinesthetic teaching. A video of the trained skills including a visualisation of the generated skill hierarchies can be viewed online[2] and is included in the supplementary material of this paper. Note that only the skills and behaviours with non-zero probabilities are shown in the hierarchies. The training of skills does not look different to the training in the basic method except for

[1]https://github.com/shangl/kukadu

[2]https://iis.uibk.ac.at/public/shangl/tro2017/hangl_roboticplaying.mp4

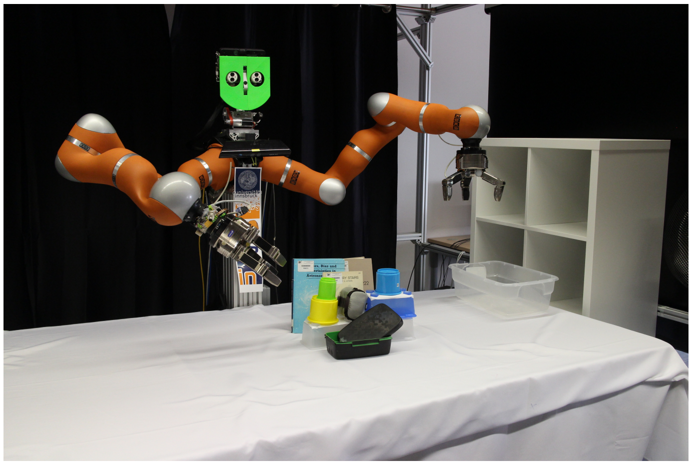


Fig. 6. Robot setting and used objects. The hardware included a Kinect, two KUKA LWR 4+ and two Schunk SDH grippers. The objects used for the trained tasks were books of different dimensions and cover types, an IKEA shelf and boxes, and selected objects of the YCB object and model set [69].

the additional execution sensing action after the performed preparation[3].

1) Simple placement: The task is to pick an object and place it in an open box on the table. The basic behaviour is a DMP that moves the grasped object to the box, where the hand is opened. In this case, the used sensing action is *weigh*, c.f. VI-A. After training the *simple grasp* / *nothing* behaviour is used if the object is not grasped / not grasped respectively.

2) Book grasping: The basic behaviour grasps a book as described in section I. The perceptual states are the four orientations of the book. After training, the robot identified *sliding* as a useful sensing action to estimate the book's rotation. The skill is trained with and without using creativity. Without creativity, the available preparatory behaviours are the *void*-behaviour, *rotate* $90°$, *rotate* $180°$, *rotate* $270°$, and *flip*. The *rotation* and *void* behaviours are used for different rotations of the book. In the creativity condition, the behaviours *rotate* $180°$ and *rotate* $270°$ are removed from the set of preparatory behaviours. The robot creates these behaviours by composing *rotate* $90°$ two / three times respectively.

3) Placing object in a box: The task is to place an object inside a box that can be closed. The basic behaviour is to grasp an object from a fixed position and drop it inside an open box. The perceptual states determine, whether the box is *open* or *closed*. After training, the robot identifies *poke* as a good sensing action. The *flip* behaviour is used to open the closed box and the *void*-behaviour is used if the box is open.

4) Complex grasping: The task is to grasp objects of different sizes. We use the *void*-behaviour as the skill's basic behaviour. This causes the robot to combine behaviours without additional input from the outside. The perceptual states correspond to small and big objects. After training, *sliding* is determined as the best sensing action. The *simple grasp* / *book grasping* behaviour is used for small / big objects respectively.

5) Shelf placement: The task is to place an object in a shelving bay, which is executed using a DMP. The robot uses the *weigh* sensing action to determine whether or not an object is already grasped. The *complex grasp* skill / *void* behaviour is used if the object is not grasped / grasped, respectively. Note that training of this skill can result in a local maximum, e.g. by choosing the behaviours *simple grasp* or *book grasp*, in particular if the reward is chosen too high.

6) Shelf alignment: The task is to push an object on a shelf towards the wall to make space for more objects. The basic behaviour is a DMP moving the hand from the left end of the shelve bay to the right end until a certain force is exceeded. As there is no object in front of the robot, all sensing actions except *no sensing* fail. The sensing action with the strongest discrimination score is *no sensing* with only one perceptual state and *shelf placement* as preparatory behaviour.

7) Tower disassembly: The task is to disassemble a stack of maximum three boxes. The basic behaviour is the *void* behaviour. The perceptual states correspond to number of boxes in the tower. Reward is given in case the tower is completely disassembled. After training, the used sensing action is *poking* to estimate four different states, i.e. height $h \in \{0, 1, 2, 3\}$. The tower cannot be removed with any single available preparatory behaviour. Instead, using the creativity mechanism, the robot generates combinations of *simple placement*, *shelf placement* and *shelf alignment* of the form given by the expression

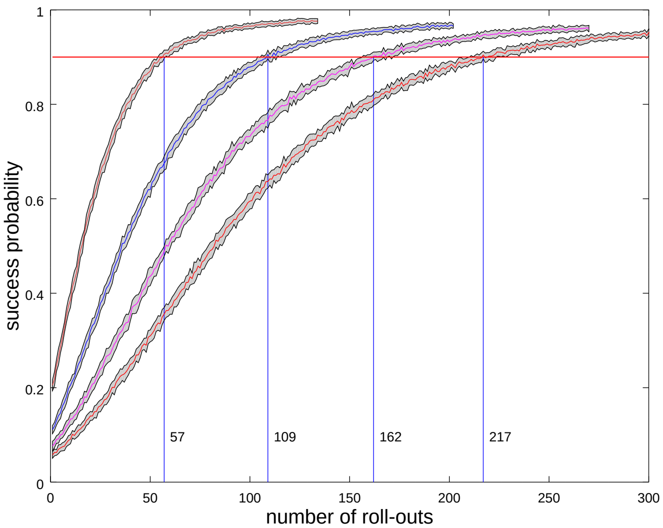
$$\text{simple placement}^* \,[\text{void} \,|\, \text{shelf placement} \,|\, \text{shelf alignment}] \tag{24}$$

C. Discussion of the real-world tasks

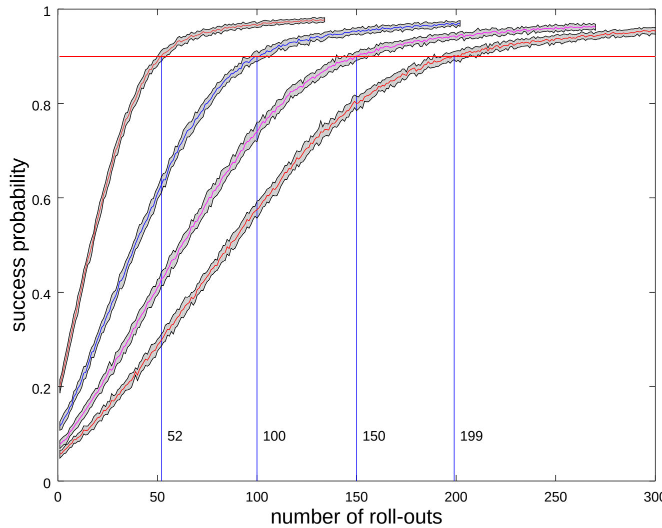
A strong advantage of model-free playing is the ability to use behaviours beyond their initial purpose. The *flip* behaviour is implemented to flip an object but is used to open the box in the *box placement* task. This holds for sensing actions as well: *sliding* is used for estimating the object size for *complex grasping* instead of the expected *pressing* from which the object size could be derived from the distance between the hands. Both sensing actions deliver a high success rate with $r_s^{\text{pressing}} \approx 0.9$ and $r_s^{\text{sliding}} \approx 1.0$. The high success rate of *sliding* is an artifact of the measurement process. The object is pushed towards the second hand until the hands get too close to each other. For small objects, the pushing stops before the finger touches the object. This produces always the same sensor data for small objects, which makes it easy to distinguish small from big objects.

In the *tower disassembly* task an important propery can be observed. The generated behaviour compositions of the form given in equation 24 only contain the skills *shelf placement* and *shelf alignment* at the end of the sequence. The reason is that these skills can only remove a box in a controlled way if only one box is left, i.e. $h = 1$. Higher towers are made to collaps because of the *complex grasping* skill, which is used by *shelf placement*. It uses *sliding* to estimate the object's size and therefore pushes the tower around. Further, which behaviour sequence is generated, depends on the subjective history of the robot, e.g. the sequences (*simple placement*, *simple placement*, *simple placement*) and (*shelf placement*, *simple placement*, *simple placement*) both yield success for $h = 3$. The autonomy of our approach can also be reduced in

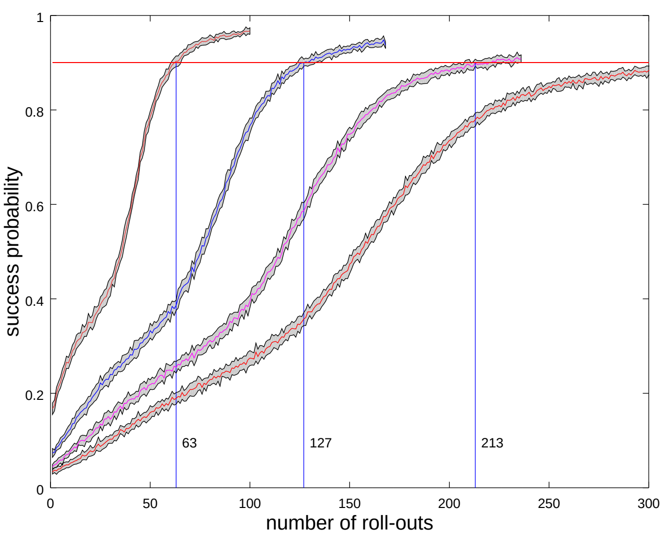
[3] https://iis.uibk.ac.at/public/shangl/iros2016/iros.mpg



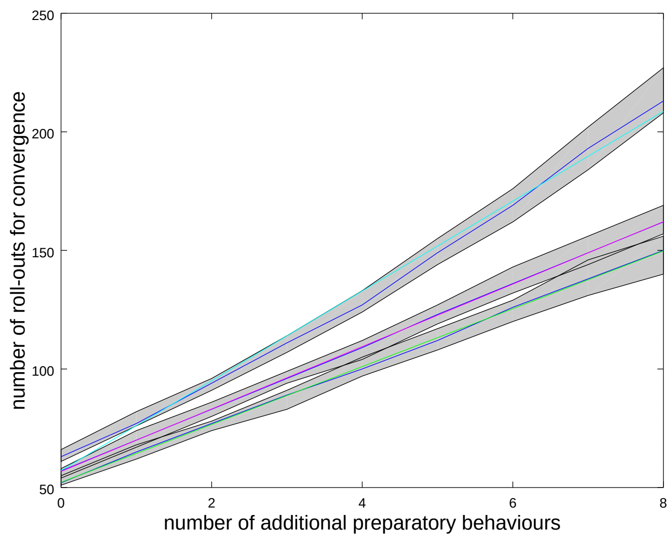
(a) Without active learning and creativity



(b) With active learning and without creativity



(c) With active learning and creativity



(d) Magenta ≡ without active learning and creativity, green ≡ with active learning and without creativity, cyan ≡ with active learning and creativity

Fig. 7. Figs. 7a – 7c: Evolution of the success rate over the number of rollouts for different numbers of preparatory behaviours. For each curve from left to right, 5 behaviours are added. The red horizontal lines denote an average success rate of 90 percent. Fig. 7d shows the number of roll-outs required to reach an average success rate of at least 90 percent for the three different versions. The straight lines show a linear fit to the measured data.

such a scenario, as several behaviours destroy the tower and require a human to prepare it again. This involves to include a human in the playing loop, in particular if the required states cannot be prepared by the robot itself.

Similarly, the active learning and creativity mechanisms do not always yield improvements. Active learning only causes a speedup if the unsolved perceptual states can be produced from solved ones, e.g. if the *closed* state is solved before the *open* state. The robot is only able to prepare the transition *closed* $\xrightarrow{flip}$ *open*. The transition *open* $\rightarrow$ *closed* requires to close the cover, which is not among the available behaviours. The creativity mechanism does not improve learning if the required behaviours are already available, e.g. in *box placement* or *shelf placement*, or cannot be composed of other behaviours. However, it helps to solve *book grasping* and *tower disassembly*.

We emphasise that the teaching of novel skills does not necessarily have to follow the typical sequence of *sensing* $\rightarrow$ *preparation* $\rightarrow$ *basic behaviour*, e.g. in *complex grasping* and *shelf alignment*. In the *complex grasping* task the basic behaviour is the *void*-behaviour, which causes the robot to coordinate different grasping procedures for small and big objects. For *shelf alignment*, the sensing stage is ommitted.

D. Simulated skill learning

Single experiments cannot be used to assess the overall convergence behaviour. We use the experiences gained in the real-world *book grasping* task to simulate the convergence behaviour. We use a success rate of 95 percent for all involved controllers. The environment is simulated with ground truth state transitions observed in the real-world experiment. For the failure cases, i.e. 5 percent of the exeuctions of each executed action, we simulate a random resulting perceptual

TABLE II
NUMBER OF ROLLOUTS REQUIRED TO CONVERGE TO A SUCCESS RATE OF AT LEAST 90 PERCENT FOR DIFFERENT NUMBERS OF BEHAVIOURS J.

J	$N_{\text{no\_ext}}$	N_{active}	N_{creative}	N_{base}
5	57	52	63	65
10	109	100	127	130
15	162	150	213	190
20	217	199	>300	260

state. The evolution of success is simulated and averaged for $N = 1000$ robots for different numbers of preparatory behaviours. The minimum number of preparatory behaviours is $J = 5$, i.e. *void*, *rotate* $90°$, *rotate* $180°$, *rotate* $270°$, *flip*. We simulate a scenario in which additional preparatory behaviours are useless, i.e. the perceptual state is not changed. In this case, the problem gets harder due to a larger set of behaviours, while the number of appropriate behaviours remains the same. In the scenario with activated creativity the agent is only provided with the behaviours *rotate* $90°$, *flip* and *void*.

The number of rollouts required to reach a success rate of at least 90 percent is given in Table II for an increasing number of behaviours J and different variants of our method ($N_{\text{no\_ext}} \equiv$ no active learning / no creativity, $N_{\text{active}} \equiv$ active learning / no creativity, $N_{\text{creative}} \equiv$ active learning / creativity, $N_{\text{base}} \equiv$ baseline). As baseline we use a policy in which every combination of perceptual states and behaviours is tried out only once, with $N_{\text{base}} = 3 * 4 * J + J$ (3 sensing actions with 4 states, 1 sensing action, i.e. *no sensing* with only one state). In general, our method converges faster than the baseline due to reducing the space strongly and ignoring irrelevant parts of the ECM. Further, the baseline method would not yield convergence in a scenario with possible execution failures as each combination is executed only once. The baseline approach also cannot solve the task in the creativity condition.

The two versions without creativity, i.e. without and with active learning, show continuous increase of the success rate in Figs. 7a and 7b. If the robot is bored, situations with a low information gain are rejected. Therefore, the version with active learning is expected to converge faster. Fig. 7d shows the number of required rollouts to reach a success rate of 90 percent for each of the three variants. The number of required rollouts is proportional to the number of available preparatory behaviours. We apply a linear fit and gain an asymptotic speed-up of $sp = 1 - \lim_{x\to\infty} \frac{k_1 x + d_1}{k_2 x + d_2} \approx 9$ percent for the variant with active learning compared to the variant without extension.

In the scenario with activated creativity the convergence behaviour is different, c.f. Fig. 7c. The success rate exhibits a slow start followed by a fast increase and a slow convergence towards 100 percent. The slow start is due to the perceptual states that would require the behaviours *rotate* $180°$ and *rotate* $270°$ which are not avaible at this point. Further, the robot cannot generate these behaviours using creativity due to initially untrained environment models. This causes the success rate to reach a preliminary plateau at around 30 to 35 percent. After this initial burn-in phase, the environment model becomes more mature and behaviour proposals are created. This causes a strong increase of the the success rate.

XI. CONCLUSION

We introduced a novel way of combining model-free and model-based reinforcement learning methods for autonomous skill acquisition. Our method acquires novel skills that work for only a narrow range of situations acquired from a human teacher, e.g. by demonstration. Previously-trained behaviours are used in a model-free RL setting in order to prepare these situations from other possibly occuring ones. This enables the robot to extend the domain of applicability of the novel skill by playing with the object. We extended the model-free approach by learning an environment model as a side product of playing. We demonstrated that the environment model can be used to improve the model-free playing in two scenarios, i.e. active learning and creative behaviour generation. In the active learning setting the robot has the choice of rejecting present situations if they are already well-known. It uses the environment model to autonomously prepare more interesting situations. Further, the environment model can be used to propose novel preparatory behaviours by concatenation of known behaviours. This allows the agent to try out complex behaviour sequences while still preserving the model-free nature of the original approach.

We evaluated our approach on a KUKA robot by solving complex manipulation tasks, e.g. complex pick-and-place operations, involving non-trivial manipulation, or tower-disassembly. We observed success statistics of the involved components and simulated the convergence behaviour in increasingly complex domains, i.e. a growing number of preparatory behaviours. We found that by active learning the number of required rollouts can be reduced by approximately 9 percent. We have shown that creative behaviour generation enables the robot to solve tasks that would not have been solvable otherwise, e.g. complex book grasping with a reduced number of preparatory behaviours or tower disassembly.

The work presented in this paper bridges the gap from plain concatenation of pre-trained behaviours to simple goal-directed planning. This can be seen as early developmental stages of a robot. We believe that a lifelong learning agent has to go through different stages of development with an increasing complexity of knowledge and improving reasoning abilities. This raises the question of how the transition to strong high-level planning systems could look like.

Our experiments show that the learning time is proportional to the number of used preparatory behaviours. This makes it efficient to learn an initial (and potentially strong) set of skills, but hard to add more skills when there is a large set of skills available already. Training more sophisticated models could help to overcome this problem. Further, in the current system, the creative behaviour generation only allows behaviour compositions resulting from plans within the same environment model, i.e. using only perceptual states of the same sensing action. The expressive power of our method could be greatly increased by allowing plans through perceptual states of different sensing actions. This could also involve multiple sensing actions at the same time including passive sensing such as vision.

Acknowledgment

The research leading to these results has received funding from the European Communitys Seventh Framework Programme FP7/20072013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 610532, Squirrel. HJB and VD acknowledge support from the Austrian Science Fund (FWF) through grant SFB FoQuS F4012.

References

[1] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

[2] W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, V. Eruhimov, T. Foote *et al.*, "Autonomous door opening and plugging in with a personal robot," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 729–736.

[3] F. J. Abu-Dakka, B. Nemec, A. Kramberger, A. G. Buch, N. Krueger, and A. Ude, "Solving peg-in-hole tasks by human demonstration and exception strategies," *Industrial Robot: An International Journal*, vol. 41, no. 6, pp. 575–584, 2014.

[4] S. Hangl, E. Ugur, S. Szedmak, A. Ude, and J. Piater, "Reactive, Task-specific Object Manipulation by Metric Reinforcement Learning," in *17th International Conference on Advanced Robotics*, 7 2015.

[5] S. Hangl, S. Krivić, P. Zech, E. Ugur, and J. Piater, "Exploiting the Environment for Object Manipulation," in *Austrian Robotics Workshop*, 5 2014.

[6] J. Piaget, *The Origins of Intelligence in Children*. New York: Norton, 1952.

[7] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, Dec 2009.

[8] A. Ferrein and G. Lakemeyer, "Logic-based robot control in highly dynamic domains," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 980 – 991, 2008, semantic Knowledge in Robotics.

[9] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for mobile robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 2020–2025.

[10] S. Hangl, E. Ugur, S. Szedmak, and J. Piater, "Robotic playing for hierarchical complex skill learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016.

[11] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[12] G. Konidaris and A. G. Barto, "Skill discovery in continuous reinforcement learning domains using skill chaining," in *Advances in neural information processing systems*, 2009, pp. 1015–1023.

[13] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, "Robot learning from demonstration by constructing skill trees," *The International Journal of Robotics Research*, p. 0278364911428653, 2011.

[14] T. R. Colin, T. Belpaeme, A. Cangelosi, and N. Hemion, "Hierarchical reinforcement learning as creative problem solving," *Robotics and Autonomous Systems*, vol. 86, pp. 196 – 206, 2016.

[15] N. D. Daw, Y. Niv, and P. Dayan, "Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control," *Nature neuroscience*, vol. 8, no. 12, pp. 1704–1711, 2005.

[16] M. Keramati, A. Dezfouli, and P. Piray, "Speed/accuracy trade-off between the habitual and the goal-directed processes," *PLoS Comput Biol*, vol. 7, no. 5, p. e1002055, 2011.

[17] L. Dollé, D. Sheynikhovich, B. Girard, R. Chavarriaga, and A. Guillot, "Path planning versus cue responding: a bio-inspired model of switching between navigation strategies," *Biological cybernetics*, vol. 103, no. 4, pp. 299–317, 2010.

[18] E. Renaudo, B. Girard, R. Chatila, and M. Khamassi, "Design of a control architecture for habit learning in robots," in *Conference on Biomimetic and Biohybrid Systems*. Springer, 2014, pp. 249–260.

[19] E. Renaudo, B. Girard, R. Chatila, and M. Khamassi, "Which criteria for autonomously shifting between goal-directed and habitual behaviors in robots?" in *Development and Learning and Epigenetic Robotics (ICDL-EpiRob), 2015 Joint IEEE International Conference on*. IEEE, 2015, pp. 254–260.

[20] K. Caluwaerts, A. Favre-Félix, M. Staffa, S. NGuyen, C. Grand, B. Girard, and M. Khamassi, "Neuro-inspired navigation strategies shifting for robots: Integration of a multiple landmark taxon strategy," in *Conference on Biomimetic and Biohybrid Systems*. Springer, 2012, pp. 62–73.

[21] K. Caluwaerts, M. Staffa, S. NGuyen, C. Grand, L. Dollé, A. Favre-Félix, B. Girard, and M. Khamassi, "A biologically inspired meta-control navigation system for the psikharpax rat robot," *Bioinspiration & biomimetics*, vol. 7, no. 2, p. 025009, 2012.

[22] A. Dezfouli and B. W. Balleine, "Habits, action sequences and reinforcement learning," *European Journal of Neuroscience*, vol. 35, no. 7, pp. 1036–1051, 2012.

[23] F. Wörgötter, C. Geib, M. Tamosiunaite, E. E. Aksoy, J. Piater, H. Xiong, A. Ude, B. Nemec, D. Kraft, N. Krger, M. Wchter, and T. Asfour, "Structural bootstrapping - a novel, generative mechanism for faster and more efficient acquisition of action-knowledge," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 140–154, June 2015.

[24] J. Weng, "Developmental robotics: Theory and experiments," *International Journal of Humanoid Robotics*, vol. 1, no. 02, pp. 199–236, 2004.

[25] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connection Science*, vol. 15, no. 4, pp. 151–190, 2003.

[26] S. Thrun, "Exploration in active learning," *Handbook of Brain Science and Neural Networks*, pp. 381–384, 1995.

[27] S. D. Whitehead, "Complexity and cooperation in q-learning," in *Proceedings of the Eighth International Workshop on Machine Learning*, 2014, pp. 363–367.

[28] L. E. Atlas, D. A. Cohn, and R. E. Ladner, "Training connectionist networks with queries and selective sampling," in *Advances in neural information processing systems*, 1990, pp. 566–573.

[29] D. A. Cohn, "Neural network exploration using optimal experiment design," *Advances in neural information processing systems*, pp. 679–679, 1994.

[30] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *Journal of artificial intelligence research*, vol. 4, no. 1, pp. 129–145, 1996.

[31] C. Chao, M. Cakmak, and A. L. Thomaz, "Transparent active learning for robots," in *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, March 2010, pp. 317–324.

[32] S. Schaal and C. G. Atkeson, "Assessing the quality of learned local models," *Advances in neural information processing systems*, pp. 160–160, 1994.

[33] L. P. Kaelbling, *Learning in embedded systems*. MIT press, 1993.

[34] S. Koenig and R. G. Simmons, "Complexity analysis of real-time reinforcement learning." in *AAAI*, R. Fikes and W. G. Lehnert, Eds. AAAI Press / The MIT Press, 1993, pp. 99–107.

[35] R. S. Sutton, "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming," in *Proceedings of the seventh international conference on machine learning*, 1990, pp. 216–224.

[36] M. Salganicoff, L. H. Ungar, and R. Bajcsy, "Active learning for vision-based robot grasping," *Machine Learning*, vol. 23, no. 2-3, pp. 251–278, 1996.

[37] A. Morales, E. Chinellato, A. H. Fagg, and A. P. del Pobil, "An active learning approach for assessing robot grasp reliability," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, Sept 2004, pp. 485–490 vol.1.

[38] O. Kroemer, R. Detry, J. Piater, and J. Peters, "Combining active learning and reactive control for robot grasping," *Robotics and Autonomous Systems*, vol. 58, no. 9, pp. 1105–1116, 2010.

[39] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[40] S. Schaal, "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics," in *Adaptive motion of animals and machines*. Springer, 2006, pp. 261–280.

[41] B. Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, no. 55-66, p. 11, 2010.

[42] P. Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 265–286, April 2007.

[43] A. Baranes and P. Y. Oudeyer, "R-iac: Robust intrinsically motivated exploration and active learning," *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 3, pp. 155–169, Oct 2009.

[44] E. Ugur and J. Piater, "Emergent structuring of interdependent affordance learning tasks using intrinsic motivation and empirical feature selection," *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[45] A. Stoytchev and R. C. Arkin, "Incorporating motivation in a hybrid robot architecture," *Journal of Advanced Computational Intelligence Vol*, vol. 8, no. 3, 2004.

[46] A. G. Barto, S. Singh, and N. Chentanez, "Intrinsically motivated learning of hierarchical collections of skills," in *Proceedings of the 3rd International Conference on Development and Learning*. Citeseer, 2004, pp. 112–19.

[47] M. Schembri, M. Mirolli, and G. Baldassarre, "Evolution and learning in an intrinsically motivated reinforcement learning robot," *Advances in Artificial Life*, pp. 294–303, 2007.

[48] J. Schmidhuber, "Formal theory of creativity, fun, and intrinsic motivation (1990 - 2010)," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 3, pp. 230–247, Sept 2010.

[49] E. Ugur and J. Piater, "Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2627–2633.

[50] G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Constructing symbolic representations for high-level planning," in *AAAI Conference on Artificial Intelligence*, 2014.

[51] G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Symbol acquisition for probabilistic high-level planning," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[52] H. J. Briegel, "On creative machines and the physical origins of freedom," *Scientific reports*, vol. 2, p. 522, 2012.

[53] H. J. Briegel and G. De las Cuevas, "Projective simulation for artificial intelligence," *Scientific Reports*, vol. 2, pp. 400 EP –, May 2012, article.

[54] A. A. Melnikov, A. Makmal, and H. J. Briegel, "Projective simulation applied to the grid-world and the mountain-car problem," *arXiv preprint arXiv:1405.5459*, 2014.

[55] J. Mautner, A. Makmal, D. Manzano, M. Tiersch, and H. J. Briegel, "Projective simulation for classical learning agents: a comprehensive investigation," *New Generation Computing*, vol. 33, no. 1, pp. 69–114, 2015.

[56] A. A. Melnikov, A. Makmal, V. Dunjko, and H. J. Briegel, "Projective simulation with generalization," *arXiv preprint arXiv:1504.02247*, 2015.

[57] M. Tiersch, E. Ganahl, and H. Briegel, "Adaptive quantum computation in changing environments using projective simulation," *Scientific reports*, vol. 5, 2015.

[58] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009.

[59] C. G. Atkeson and S. Schaal, "Learning tasks from a single demonstration," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 2, Apr 1997, pp. 1706–1712 vol.2.

[60] T. Asfour, P. Azad, F. Gyarfas, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *International Journal of Humanoid Robotics*, vol. 05, no. 02, pp. 183–202, 2008.

[61] M. Lopes, F. S. Melo, and L. Montesano, "Affordance-based imitation learning in robots," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2007, pp. 1015–1021.

[62] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[63] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[64] S. Krivic, E. Ugur, and J. Piater, "A Robust Pushing Skill For Object Delivery Between Obstacles," in *Conference on Automation Science and Engineering*, 08 2016, fort Worth, Texas.

[65] D. Omrcen, C. Boge, T. Asfour, A. Ude, and R. Dillmann, "Autonomous acquisition of pushing actions to support object grasping with a humanoid robot," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, Dec 2009, pp. 277–283.

[66] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

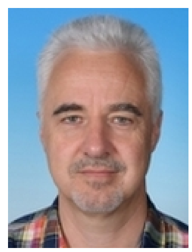
[67] Y. Li, "Hybrid control approach to the peg-in hole problem," *IEEE Robotics Automation Magazine*, vol. 4, no. 2, pp. 52–60, Jun 1997.

[68] D. E. Whitney, "Historical perspective and state of the art in robot force control," *The International Journal of Robotics Research*, vol. 6, no. 1, pp. 3–14, 1987.

[69] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *Advanced Robotics (ICAR), 2015 International Conference on*. IEEE, 2015, pp. 510–517.

Simon Hangl is a PhD student at the intelligent and interactive systems (IIS) group of the University of Innsbruck, where he also received his MSc degree in Computer Science. Before starting his position as University Assistant at IIS, he worked as researcher at the Semantic Technology Institute (STI) Innsbruck. He worked in 4 EU-FP7 projects in the areas of semantic technology and robotics. He is interested in developmental and cognitive robotics and complex robotic object manipulation.

Vedran Dunjko received the Ph.D. degree in physics from Heriot-Watt University, Edinburgh, U.K., in 2012, focused on problems in quantum cryptography, specifically blind quantum computing, and quantum digital signatures. Following a one-year post-doctoral position with the School of Informatics, University of Edinburgh, in 2013, he then moved to the Institute of Theoretical Physics, University of Innsbruck, Austria, where he is currently a Post-Doctoral Researcher. He has recently been involved in the problems of artificial intelligence and quantum machine learning.



Hans J. Briegel received the Ph.D. degree (doctorate) in physics in 1994 and the Habilitation degree in theoretical physics in 2002 from the Ludwig-Maximilians-University of Munich. He held post-doctoral positions with Texas A&M and Harvard University. He has been a Full Professor of Theoretical Physics with the University of Innsbruck since 2003 and a Research Director with the Institute of Quantum Optics and Quantum Information of the Austrian Academy of Sciences from 2003 to 2014. His main field of research is quantum information and quantum optics where he has authored and co-authored papers on a wide range of topics, including work on noise reduction and microscopic lasers, quantum repeaters for long-distance quantum communication, entanglement and cluster states, and measurement-based quantum computers. His recent research has focussed on physical models for classical and quantum machine learning, artificial intelligence, and the problem of agency. HJ Briegel is also with the Department of Philosophy, University of Konstanz, Germany.

Justus Piater is a professor of computer science at the University of Innsbruck, Austria, where he leads the Intelligent and Interactive Systems group. He holds a M.Sc. degree from the University of Magdeburg, Germany, and M.Sc. and Ph.D. degrees from the University of Massachusetts Amherst, USA, all in computer science. Before joining the University of Innsbruck in 2010, he was a visiting researcher at the Max Planck Institute for Biological Cybernetics in Tübingen, Germany, a professor of computer science at the University of Lige, Belgium, and a Marie-Curie research fellow at GRAVIR-IMAG, INRIA Rhne-Alpes, France. His research interests focus on visual perception, learning and inference in sensorimotor systems. He has published more than 150 papers in international journals and conferences, several of which have received best-paper awards, and currently serves as Associate Editor of the IEEE Transactions on Robotics.