MPR-RL: Multi-Prior Regularized Reinforcement Learning for Knowledge Transfer

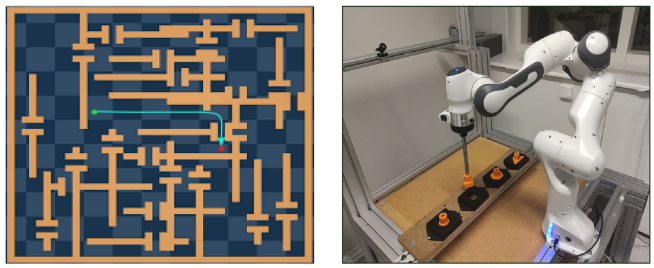
Quantao Yang, Johannes A. Stork, and Todor Stoyanov, *Member, IEEE*

***Abstract*—In manufacturing, assembly tasks have been a challenge for learning algorithms due to variant dynamics of different environments. Reinforcement learning (RL) is a promising framework to automatically learn these tasks, yet it is still not easy to apply a learned policy or skill, that is the ability of solving a task, to a similar environment even if the deployment conditions are only slightly different. In this letter, we address the challenge of transferring knowledge within a family of similar tasks by leveraging multiple skill priors. We propose to learn prior distribution over the specific skill required to accomplish each task and compose the family of skill priors to guide learning the policy for a new task by comparing the similarity between the target task and the prior ones. Our method learns a latent action space representing the skill embedding from demonstrated trajectories for each prior task. We have evaluated our method on a task in simulation and a set of peg-in-hole insertion tasks and demonstrate better generalization to new tasks that have never been encountered during training. Our Multi-Prior Regularized RL (MPR-RL) method is deployed directly on a real world Franka Panda arm, requiring only a set of demonstrated trajectories from similar, but crucially not identical, problem instances.**

***Index Terms*—Machine learning for robot control, reinforcement learning, transfer learning.**

I. Introduction

HUMANS are adept in transferring a learned skill, that is the ability of solving a task, to a new similar task efficiently—for example humans are able to grasp an unseen object by prior experience. However, most state-of-the-art reinforcement learning (RL) methods learn every task from scratch and it is not easy to apply the learned policy to a new problem, even if it is a very similar one [1]. Consequently, millions of new interactions with different environments can be required to solve variant tasks, which is infeasible for a real robot system. Also, retraining is resource and time consuming, meanwhile sample collection in a new environment is costly and repetitive. Therefore, it is imperative to address these problems in order to apply RL in similar environments directly on real physical robots.



(a) (b)

Fig. 1. An illustration of the experimental setup for two robot systems: (a) 2D maze navigation task and (b) a 7-DOF redundant Franka Panda for peg-in-hole tasks. We train multiple skill priors for a family of problems from demonstrated trajectories for each task.

The state-of-the-art methods require policy training in simulation to prevent undesired behavior and later domain transfer, or guided policy search for single skills in a family of similar problems [2]–[4]. The successful deployment of simulation-to-reality methods [5] requires that the simulation is close enough to the physical system. However, for real world robotic applications, the dynamics encountered in the deployment phase are often radically different from those seen during training, which might lead to the failure of transferring knowledge. Our prior work [6] utilized a framework for learning latent action spaces for RL agents from demonstrated trajectories [7] and connected it to a variable impedance Cartesian space controller, allowing us to learn contact-rich tasks safely and efficiently. However, the method [6] requires demonstration from an expert or expert policy in the precise domain we are solving.

In this work, we consider the problem of transferring knowledge within a family of similar tasks. Our fundamental assumption is that we are presented with a family of problems, formalized as Markov Decision Processes (MDPs) that all share the same state and action spaces. Crucially however, we allow for members of the family to exhibit different transition dynamics. Informally, our assumption is that while transition probabilities are different, they may be correlated or overlapping for parts of the state space. We then propose a method — Multi-Prior Regularized RL (MPR-RL) — that leverages prior experience collected on a subset of the problems in the MDP family to efficiently learn a policy on a new, previously unseen problem

Manuscript received 24 February 2022; accepted 12 June 2022. Date of publication 22 June 2022; date of current version 29 June 2022. This letter was recommended for publication by Associate Editor D. Sadigh and Editor J. Kober upon evaluation of the reviewers' comments. This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation. *(Corresponding author: Quantao Yang.)*

Quantao Yang and Johannes A. Stork are with the Autonomous Mobile Manipulation lab at the Center for Applied Autonomous Sensor Systems (AASS), Örebro University, 702 81 Örebro, Sweden (e-mail: quantao.yang@oru.se; johannesandreas.stork@oru.se).

Todor Stoyanov is with the Autonomous Mobile Manipulation lab at the Center for Applied Autonomous Sensor Systems (AASS), Örebro University, 702 81 Örebro, Sweden, and also with the Department of Computing and Software, McMaster University, Hamilton, ON L8S 4L8, Canada (e-mail: todor.stoyanov@oru.se).

Digital Object Identifier 10.1109/LRA.2022.3184805

from the same family. Our approach learns prior distributions over the specific skill for each task and composes a family of skill priors to guide learning the policy in a new environment. We have evaluated our method on a 2D maze navigation task shown in Fig. 1(a) and contact-rich peg-in-hole tasks shown in Fig. 1(b). We show that our MPR-RL method can guide the policy learning over similar problems and that the composition of multiple skill priors accelerates training the RL policy.

The main contributions of this letter are: (1) we learn multiple skill priors for a family of similar problems from demonstrated trajectories for each prior task; (2) we propose to learn an adaptive strategy for composing multiple priors to guide the policy learning on a new problem by comparing the similarity between the target task and the prior ones; (3) we evaluate our method on a maze navigation environment and a number of peg-in-hole task variants with a Franka Panda arm and demonstrate that our MPR-RL method can be generalized to a new environment and be deployed directly on the real robot.

II. Related Work

A. Knowledge Transfer Learning

Transfer learning has been a promising technique to leverage prior experience to improve learning efficiency and generalization ability [8]–[10]. Recently, transferring knowledge within a family of Markov decision processes (MDPs) has been investigated. In [11], variational policy embedding (VPE) was proposed to learn a master policy to accelerate the adaptation to new members of the family. Similar to [11], we transfer knowledge betweeen MDPs with the same state spaces and action spaces. We also use a low-dimensional latent space to represent the behavior of the policy. A method [12] decomposes multi-task problems to handle novel task combinations by learning modular neural network policies that are shared across all tasks or robots. Gupta *et al.* proposed invariant feature spaces to transfer skills from one agent to another [13]. In their case, the agents have prior knowledge about each other and have different state spaces and action spaces, while our method focuses on transferring knowledge between a family of tasks for one agent. Yet their work is still limited to tasks or robots in simulation. In comparison, our MPR-RL method can be deployed on the real system directly. Another approach for transferring knowledge is domain randomization [14]–[17] that learns one policy feasible to variant MDPs with different dynamics. However, a variety of simulated environments with randomized properties are required to enable the policy to adapt to novel MDPs. Our approach utilizes prior knowledge for each task and transfers knowledge to a novel MDP.

B. Meta Reinforcement Learning

Most modern RL approaches learn every task from scratch and consequently the generalization of RL applied to robotics is still challenging. Meta RL approaches essentially aim to learn a good space that can quickly be adapted in order to learn a new skill. Model-agnostic meta learning (MAML) [18] has demonstrated good performance on continuous control tasks in simulation, which explicitly updates policy parameters after a few steps. Several methods [19], [20] have been proposed to improve the generalization performance of MAML, however applications to real robot systems still remain challenging.

To deploy meta RL in the real world, a dynamics model prior was trained to accelerate the policy adaption to the local context in [21]. Another recent work [22] combines gradient-based meta learning with latent variable generative models to adapt the meta RL policy to large variations both in simulation and on real systems. Although these methods achieve success in real world environments, they are struggling in solving contact-rich tasks. In [23], a latent structure of each task is learned to solve a family of contact-rich insertion tasks by randomizing task parameters. An off-policy meta RL algorithm, probabilistic embeddings for actor-critic meta RL (PEARL) [24], is leveraged to embed each insertion task into the latent space. Despite the recent advances in the field, these methods still require the RL model to be trained in simulation and then deployed on the real system. This might cause inconsistency between simulation and reality during the real world deployment. In this letter, instead of learning a meta model, we compose multiple skill priors to solve a target task by using the knowledge from a family of similar problems.

C. Skill Prior Reinforcement Learning

Prior knowledge of the task can significantly improve the RL learning performance and generalization capabilities. One approach for leveraging prior knowledge is to learn a deep latent space of skills and a prior distribution on skills within that space, using offline experience [7]. This skill prior based RL performs more efficiently over long-horizon tasks, but still requires many interactions to learn a new task. Skill-based learning with demonstrations [25] was proposed to accelerate the learning of a new task by using a small number of task-specific demonstrations. The work [6] extended the framework in [7] by connecting it to a variable impedance Cartesian space controller and it allows us to learn contact-rich tasks directly on the real robot. The key idea of these works is to learn a prior over skills along with a skill library to guide exploration in skill space and enable efficient downstream learning, even with large skill spaces. In contrast, our MPR-RL method focuses on short-horizon and more complex tasks.

Singh *et al.* proposed to pre-train behavioral priors [26] from a diverse multi-task dataset to accelerate learning new skills. However, training behavioral priors needs a wide range of previously seen tasks to achieve robustness. A recent work [27] proposes a bottom-up approach to learning a set of reusable skills from multi-task, multi-sensory demonstrations and use these skills to synthesize long-horizon robot behaviors. In comparison, our approach is able to learn multiple skill priors for a family of problems and be adapted directly on the target physical system by consuming a fraction of the interaction samples.

III. Approach

Our approach to transferring knowledge for RL is based on exploiting prior knowledge from demonstrations for learning a

policy for a new task. The process is composed of two distinct phases: a prior learning phase and a policy learning phase. In the former we learn a latent skill space, along with a set of guiding distributions (priors) for each demonstrated task; in the latter, we utilize the learned priors to speed up policy learning. For this, we regularize the RL objective with a relative entropy term based on the learned skill priors. Below we formularize the problem setting, then we detail our skill prior learning, derive our multi-prior regularization method and finally propose a method to determine skill prior contribution during the task learning phase.

A. Problem Formulation

An RL agent acts according to a policy distribution $\pi_\theta(a|s)$ with states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$. The agent is trained based on a reward signal $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and aims to maximize the expected return:

$$G(\theta) = \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t r(s_t, a_t)\right], \tag{1}$$

where T is the episode horizon, τ is the state-action trajectory and $\gamma^t \in (0, 1]$ is the discount rate at time t.

We consider a family of tasks where each task is formulated as a Markov decision process (MDP) defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \rho, \gamma)$ of states, actions, transition probability, reward, initial state distribution, and discount factor. A family of MDPs, $\mathcal{M}$, share the same state space and action space, while the dynamics for these MDPs and transition probabilities are different.

We assume access to a dataset D of demonstrated trajectories $\tau_i = \{(s_0, a_0), \ldots, (s_{T_i}, a_{T_i})\}$ for each task. We aim to leverage these trajectories to learn a skill prior $p_i(a_t|s_t)$ for each specific MDP M_i. Our objective is then to learn a policy $\pi_\theta(a|s)$ with parameter θ that maximizes the sum of rewards $G(\theta)$ for a new MDP M_{new} by leveraging the prior experience contained in the dataset D.

B. Skill Prior Learning

In the prior learning phase, we use demonstrated trajectories τ_i to learn distributions over skill priors. We use a variational autoencoder (VAE) [28] model to learn a low-dimensional skill latent space $\mathcal{Z}$ from a dataset of pre-collected trajectories. The VAE model consists of a skill encoder $q(z|\boldsymbol{a})$ that outputs the latent representation z of a skill and a decoder $p_{dec}(\boldsymbol{a}|z)$ that predicts a sequence of low-level actions $\boldsymbol{a} = \{a_t, \cdots, a_{t+H-1}\}$ that the skill embedding z represents, where $H \in \mathbb{N}^+$ is the action horizon. As described in [7], the skill prior model $p_{\boldsymbol{a}}(z|s_t)$ is used to generate a prior distribution over the latent space $\mathcal{Z}$ based on the state s_t. This distribution serves as guidance for the policy to determine which skills are worth exploring. Following [29] we train the VAE by optimizing the evidence lower bound (ELBO):

$$\log p(\boldsymbol{a}) \geq \mathbb{E}_q[\log p(\boldsymbol{a}|z) - \beta\left(\log q(z|\boldsymbol{a}) - \log p(z)\right)], \tag{2}$$

where β is a hyperparameter used to tune the regularization term.

In skill prior RL (SPiRL) [7], the learned skill prior is leveraged to guide learning a high-level policy $\pi_\theta(z|s)$ by introducing an entropy term. They propose to replace the entropy term of Soft Actor-Critic (SAC) [30] with the negated KL divergence between the policy and the prior. Similarly, our method uses the embedding space $\mathcal{Z}$ as the RL action space as shown in Fig. 2 and the policy is over skills $\pi_\theta(z|s)$, for ease of notation, we will still use a as the action in Section III-C.

C. Multi-Prior Regularized Reinforcement Learning

Our MPR-RL can be derived in the following way. In maximum entropy RL [30], the RL objective is regularized with the policy entropy:

$$J(\theta) = \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t r(s_t, a_t) + \alpha\Gamma_t\right], \tag{3}$$

where $\Gamma = \mathcal{H}(\pi_\theta(a_t|s_t))$ is the policy entropy term and α is the temperature parameter which decreases over time. This means that the policy is initially incentivized to explore more widely [30]. The regularization term Γ can be rewritten as the relative entropy between the policy and the uniform distribution over actions $U(a_t)$:

$$\Gamma_t = -D_{\mathrm{KL}}(\pi_\theta(a_t|s_t), U(a_t)). \tag{4}$$

This formulation can be exploited to guide the learning process with a learned non-uniform distribution [7]:

$$\Gamma_t = -D_{\mathrm{KL}}(\pi_\theta(a_t|s_t), p(a_t|s_t)). \tag{5}$$

This means that the policy is initially incentivized to explore according to the learned distribution $p(a_t|s_t)$ which can aid the learning process if $p(a_t|s_t)$ is similar to the optimal policy in the task. For instance, this is the case when it is learned from demonstrations of the same task. Following [7], we also call the distribution $p(a_t|s_t)$ skill prior.

Using only one skill prior limits the method to policy learning in the same task as where the skill prior was learned. For this reason, we extend this approach from one learned skill prior to several skill priors learned in different tasks. To this end, we regularize the RL objective with a weighted sum of relative entropies:

$$\Gamma_t = -\sum_{i=1}^{m}\omega_i D_{\mathrm{KL}}(\pi_\theta(a_t|s_t), p_i(a_t|s_t)), \tag{6}$$

where ω_i is an adaptive weight and $\sum_{i=1}^{m}\omega_i = 1$, and $p_i(a_t|s_t)$ are skill priors from different tasks of the family. This means that the policy is initially incentivized to explore according to a mixture of different skill priors depending on the weight factors. While there are many options, in the next section we explain how we define the weight factors based on the current transition.

D. Learning Prior Weight

The key idea behind our method is to give high weight to those regularizing priors that come from similar tasks. We assume that the tasks only differ in dynamics, therefore we predict the weights based on the transitions. To this end, we train a prior

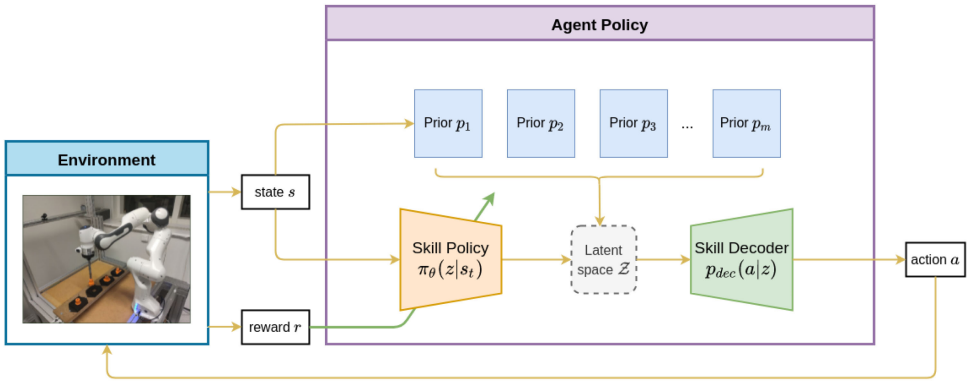


Fig. 2. MPR-RL framework: we pre-train multiple skill priors for a family of similar MDPs. Once skill priors $p_i(z|s_t)$ and the skill decoder $p_{dec}(a|z)$ block are learned, the skill policy $\pi_\theta(z|s_t)$ is trained by composing multiple prior divergences for a target task to generate embedding action z that can be decoded into a sequence of real action commands.

predictor $\Omega(s_t, a_t, s_{t+1})$ to discriminate between state-action-state transitions generated by each member M_i in our family of MDPs. Intuitively, when the transitions in all members are very similar, our model Ω predicts uniform weights. Conversely, in cases when members of the family differ a lot, Ω can clearly identify the correct problem setting.

We design a prior predictor $\Omega(s_t, a_t, s_{t+1})$ as a feed-forward neural network to determine each prior weight ω_i. The training of the prior predictor is treated as a supervised learning problem using the demonstrated datasets for a family of accessible MDPs. In contrast to constant weights, this allows different skill priors to guide exploration in different parts of the state space. As a consequence, knowledge from different tasks can be transferred for learning in a new environment.

In our case, we represent predictor Ω as a multi-layer neural network and assume the prior divergence weights are multinomial probabilities that sum up to 1. Therefore, we use the softmax function as the activation function in the output layer of neural network. Once learned, the predictor is used as an approximator that outputs the adaptive weight ω_i for each skill prior.

E. MPR-RL Algorithm

Assuming that the agent has access to prior knowledge of m similar MDPs, we aim to transfer the learned knowledge to a new MDP. We regularize the policy with the negated KL divergences between the policy and each skill prior. The policy π_θ for a new MDP M_{new} can be learned by optimizing the objective function. We propose to utilize a learned predictor $\Omega(s_t, a_t, s_{t+1})$ to determine the importance of each prior for a new task. The framework of our MPR-RL is illustrated in Fig. 2. We learn a low-dimensional latent representation $z \in \mathcal{Z}$ for RL action. Multiple priors for a family of MDPs are pre-trained to estimate the KL divergence with a skill policy $\pi_\theta(z|s_t)$. The policy $\pi_\theta(z|s_t)$ over the latent action space is trained to output embeddings that are decoded into sequences of real actions by the skill decoder $p_{dec}(\boldsymbol{a}|z)$.

Our MPR-RL approach is summarized in Algorithm 1. We modify the Soft Actor-Critic (SAC) by adding the sum of multiple KL divergences between the skill policy and pre-trained skill priors. During each gradient step, we predict the prior divergence weights and compose all priors to estimate the policy entropy. The temperature parameter α in equation (3) trades off the cumulative reward and the regularization term. We tune the regularization weight α automatically by defining a target divergence δ between policy and action priors. We formulate it as a constraint optimization problem:

$$\max \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t r(s_t, a_t)\right],$$

Algorithm 1: MPR-RL

Input : Learning rates $\lambda_\pi, \lambda_Q, \lambda_\alpha$, target divergence δ, target update rate τ, priors p_i and prior predictor Ω

Output: Trained policy $\pi_\theta(z_t|s_t)$

Initialize the policy $\pi_\theta(z_t|s_t)$, critic $Q_\phi(s_t, z_t)$, target network $Q_{\bar{\phi}}(s_t, z_t)$ and initialize replay buffer $\mathcal{B}$
for *each episode=1,M* **do**
  for *each step=1,N* **do**
    Select high-level action $z_t \sim \pi_\theta(z_t|s_t)$
    Execute action z_t and receive reward r_t
    Store transition tuple $(s_t,\ z_t,\ r_t,\ s_{t+1})$ in $\mathcal{B}$
  end
  for *each gradient step* **do**
    $\boldsymbol{\omega} = \Omega(s_t, a_t, s_{t+1})$
    $\bar{Q} = r + \gamma[Q_{\bar{\phi}}(s_{t+1}, \pi_\theta(z_{t+1}|s_{t+1})) - \alpha \sum_{i=0}^{m} \omega_i D_{\mathrm{KL}}(\pi_\theta(z_{t+1}|s_{t+1}), p_i(z_{t+1}|s_{t+1}))]$
    $\theta \leftarrow \theta - \lambda_\pi \nabla_\theta [Q_\phi - \alpha \sum_{i=0}^{m} \omega_i D_{\mathrm{KL}}(\pi_\theta, p_i)]$
    $\phi \leftarrow \phi - \lambda_Q \nabla_\phi [\frac{1}{2}(Q_\phi - \bar{Q})^2]$
    $\alpha \leftarrow \alpha - \lambda_\alpha \nabla_\alpha [\alpha(\sum_{i=0}^{m} \omega_i D_{\mathrm{KL}}(\pi_\theta, p_i) - \delta)]$
    $\bar{\phi} \leftarrow \tau\phi + (1-\tau)\bar{\phi}$
  end
end
return the trained policy $\pi_\theta(z_t|s_t)$

$$\text{s.t.} \sum_{i=0}^{m} \omega_i D_{\text{KL}}(\pi_\theta, p_i) \leq \delta, \tag{7}$$

where p_i is the prior model for each MDP. This leads to the update rule for α:

$$\alpha \leftarrow \alpha - \lambda_\alpha \nabla_\alpha \left[\alpha \left(\sum_{i=0}^{m} \omega_i D_{\text{KL}}(\pi_\theta, p_i) - \delta \right) \right], \tag{8}$$

where λ_α is the learning rate. The regularization weight α controls the stochasticity of the optimal policy. Due to the constraint term, α should decrease over time if the knowledge from a family of MDPs is transferred to a new MDP successfully.

IV. EVALUATION

We evaluate our approach on a simulated maze navigation environment [31] and a contact-rich peg-in-hole environment with a real Franka Panda arm. The goals of all experiments are three-fold: (1) to evaluate the performance of our MPR-RL on a novel MDP; (2) to test whether the prior predictor accelerates the learning process for a novel MDP; (3) to investigate whether our MPR-RL is able to achieve better generalization ability compared to the baseline. In the following, we first describe the experimental setup in IV-A, then present experiment results for the simulated task in IV-B and the real robot tasks in IV-C, lastly show ablation experiments in IV-D.

A. Experimental Setup

For training the skill embedding variable, the encoder and decoder of the VAE model are implemented as a long short-term memory (LSTM) of 128 hidden units to represent a sequence of robot commands. Similar to [7], each skill prior is represented as a 5-layer fully-connected network with 128 hidden units per layer. For the maze navigation environment, a convolutional network is used to extract the image feature concatenated with the agent proprioceptive state. The prior predictor is modeled as a 3-layer neural network with 32 units per layer to avoid overfitting.

We use MPR-RL in Algorithm 1 to learn a latent action with RL discount factor 0.99 and batch size 128. The latent variable z is embedded as the Gaussian posterior distribution of different dimensions. We use Adam as the optimizer, with an initial learning rate of 1e-3. We tuned some hyperparameters in our experiments and chose the regularization weight β in equation (2) as 5e-5. During RL policy training, all pre-trained models of multiple skill priors for different MDPs were kept fixed.

We evaluate the performance of our MPR-RL using adaptive weights and compare with several baseline methods: (1) MPR-RL hard-max weight, where only the skill prior with the maximum task likelihood under the transition is used, (2) MPR-RL uniform weight, (3) SPiRL without (w/o) prior knowledge from the target MDP, (4) Soft Actor-Critic (SAC), and (5) Behavioral Cloning with SAC (BC+SAC). When SPiRL has full access to the demonstration data from the target task, it is a baseline that is provided as a target to try to reach. We compare all methods' access to the demonstration data from the target or similar task in Table I.

TABLE I
COMPARISON OF DIFFERENT METHODS

Method	Access to similar task	Access to target task	Online learning	Result
MPR-RL adaptive	✓	✗	✓	success
MPR-RL hard-max	✓	✗	✓	success
MPR-RL uniform	✓	✗	✓	fail
SPiRL	✗	✓	✓	success
SPiRL w/o knowledge	✓	✗	✓	fail
SAC	✗	✗	✓	fail
BC+SAC	✗	✓	✓	fail

TABLE II
PARAMETERS FOR DIFFERENT MAZE ENVIRONMENTS

Parameter	Minimum	Maximum
damping ζ	0.4	1.6
x linear friction μ_x	0.2	1.2
y linear friction μ_y	0.2	1.2

B. Simulated Maze Navigation Task

We evaluated our method in a simulated 2D maze navigation task based on the D4RL environment [31]. The goal is to navigate the agent from a fixed start point to target locations. The RL state space is represented as the combination of agent position, velocity and a 32×32 pixel view centered around the agent. The action space is chosen as the 2D velocities of the agent. A sparse binary reward is received when the agent moves in close vicinity to the goal. For each maze navigation MDP, we collected 20 K goal-reaching trajectories to train the prior model and the prior predictor is learned using the combined 60 K trajectories of three MDPs (not the target MDP). For SPiRL, we use 85 K goal-reaching trajectories as in [7] to train the skill prior model and the latent space.

To vary the dynamics of the maze environment, some parameters are randomized as listed in Table II. We consider the damping ζ, x linear friction μ_x and y linear friction μ_y. We generated datasets from three maze MDPs with different dynamic parameters.

Fig. 3(a) shows the learning curves of our MPR-RL and the baselines. In all experiments, both our MPR-RL with adaptive weight and SPiRL succeed in learning to reach the goal. Our method does not need experience from the novel target MDP, while SPiRL fails to transfer prior knowledge from a family of MDPs to a novel one. Also, the results of MPR-RL with adaptive weight and uniform weight indicate that weighting divergences for optimizing the objective function in equation (6) has critical influence on transferring knowledge between a family of MDPs. It can also be seen that the variance of hard-max method is larger, which means that its performance is not stable as our adaptive weighting. Without prior knowledge from the target task, SPiRL fails to learn a solution policy. SAC from scratch and BC + SAC cannot finish the task.

The exploration behaviors of four methods are shown in Fig. 4. Our MPR-RL with adaptive weight is able to learn a

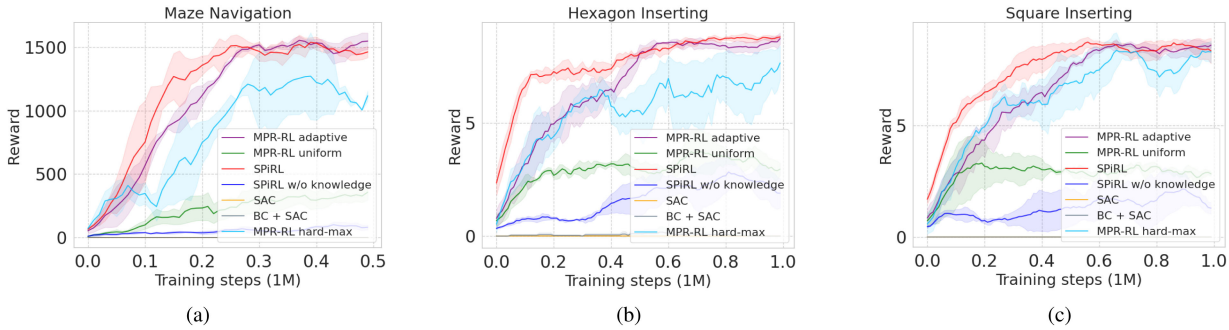


Fig. 3. Comparing learning curves of MPR-RL with baseline methods: (a) rewards for maze navigation task; (b) rewards for inserting hexagon peg experiments; and (c) rewards for inserting square peg experiments. Our MPR-RL method converges to good performance when used with adaptive prior weighting. Without prior knowledge from the target task, SPiRL fails to learn a solution policy. SAC from scratch and BC + SAC cannot finish the task. Shaded areas show standard deviation for three seeds.

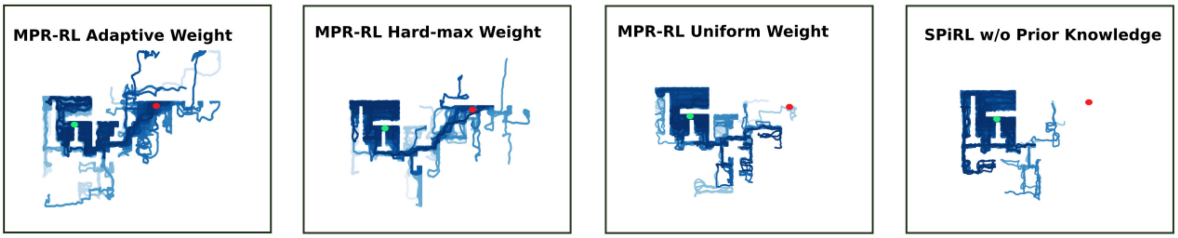


Fig. 4. Exploration behavior of our MPR-RL compared with baseline methods. The start and goal positions are depicted in green and red respectively. Darker blue indicates recent exploration areas. Our MPR-RL method is able to explore the vicinity in a novel MDP by transferring knowledge from a family of similar MDPs. The prior predictor helps the policy to explore in the goal point area. In comparison, SPiRL cannot learn the policy if no prior knowledge of the new MDP is available.

navigation policy in a new MDP successfully by transferring prior knowledge, while if no predictor Ω (uniform weight) is used the agent concentrates on exploring other area far away from the goal. It can be observed that hard-max weighting scheme decreases the agent's exploration ability. For SPiRL using priors from the family of MDPs except the target one, the agent explores a confined area of the maze around the start point and fails to reach the goal. We conclude that SPiRL needs prior experience from the target MDP to achieve similar performance as our method. We clarify our MPR-RL is more sample efficient because when we adapt our policy to a new MDP environment or even other systems from the same MDP family, it is not necessary to collect the samples in the new MDP for MPR-RL, while it is for SPiRL.

C. Real Robot Experiments

In the contact-rich peg-in-hole tasks, we use four different shapes as a family of MDPs: circular, hexagon, square and triangular as shown in Fig. 1(b). The different shapes induce different contact dynamics and thus modify the transition probabilities of the MDPs in the family. We test two cases: (1) inserting hexagon peg by using prior knowledge from circular, square and triangular ones; (2) inserting square peg by using prior knowledge from circular, hexagon and triangular ones. These two cases offer the possibility to interpolate between some of the other shapes.

To implement contact-rich tasks, we use a Cartesian impedance controller. For a robot with k joints, the observation vector s_t is composed of: (1) joint positions $\boldsymbol{q} \in \mathbb{R}^k$ and joint velocities $\dot{\boldsymbol{q}} \in \mathbb{R}^k$, (2) end-effector position offset $\boldsymbol{e} \in \mathbb{R}^3$ and rotation θ_z in the z direction, and (3) the environment contact force along the z direction $F_{ext} \in \mathbb{R}$.

To train the skill prior in advance, we collect 100 insertion trajectories for each MDP using a finite state machine that divides each trajectory into downward reaching, spiral motion alignment and insertion. The purpose of spiral alignment is to search for the target hole in a large area. In our case the Archimedean spiral motion is defined in polar coordinates (r_p, φ):

$$r_p = b_0 + b_1 \varphi, \tag{9}$$

where b_0 describes how far away from the origin the spiral should start and b_1 represents the distance between each turn of the spiral. We also use a $\sin$ function to control the rotation along the z axis during spiral motion. The reward function is negatively proportional to the distance between the end-effector and the hole.

To train skill prior RL on the real robot directly, the system stiffness term $\boldsymbol{K} \in \mathbb{R}^{6\times6}$ is incorporated into the agent action. Therefore, we extend the policy action as the combination of end-effector pose $\boldsymbol{\xi} \in SE(3)$ in Cartesian space and variable stiffness matrix $\boldsymbol{K}$. Stiffness matrix $\boldsymbol{K}$ contains 6-dimensional end-effector stiffness coefficients. One extra null-space stiffness coeffcient for the redundant robot is set as a constant value.

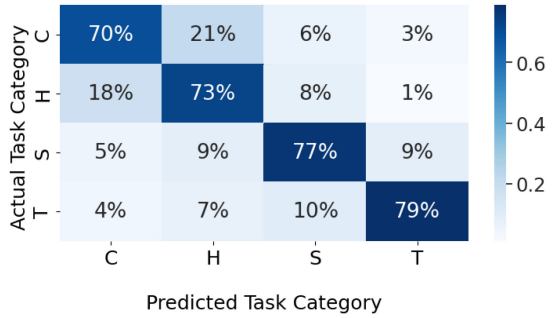


Fig. 5. Confusion matrix of visualizing the similarity among all peg-in-hole tasks. Through predicting transition samples from four tasks, the prior predictor estimates the similarity accurately. C, H, S and T stand for circular, hexagon, square and triangular respectively.

Our 8-dimensional action space is thus composed of: (1) end-effector translations $\boldsymbol{x} \in \mathbb{R}^3$ in Cartesian space, (2) rotational angle $\theta_z \in \mathbb{R}$ around the z axis, and (3) the diagonal coefficients $\boldsymbol{k} \in \mathbb{R}^4$ that determine the variable stiffness matrix $\boldsymbol{K}$ for the corresponding four Cartesian components.

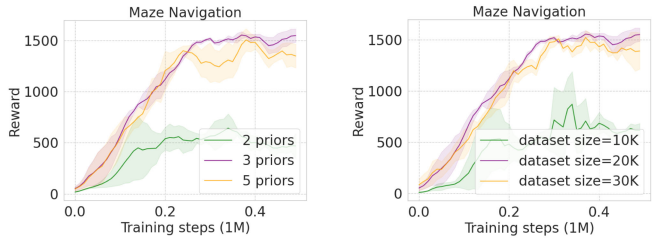
We use demonstrated trajectories to train the skill prior for each task and trained the prior predictor with the labeled trajectories from all tasks. We sample 100 transitions from each task and discriminate the task category by the prior predictor. We evaluate the performance of the prior predictor by calculating a confusion matrix of different tasks. In Fig. 5 it can be seen that there is a sharp distinction between the triangular shape and other shapes, while the circular peg is more similar to the hexagon. We hypothesize that the triangular shape leads to more complex dynamics during insertion interactions. We note that due to the lightweight design of the prior predictor, discrimination accuracy is overall low at around 75%. Nevertheless, this is sufficient to adequately weight the priors associated with similar tasks in the MDP family.

Fig. 3(b) and Fig. 3(c) show the learning curves of our MPR-RL method and baselines over the real world hexagon and square peg-in-hole tasks respectively. Each method is trained using three different seeds. We can see that SPiRL with prior knowledge of the current MDP learns slightly faster, which is expected. MPR-RL policy with adaptive weight is able to learn to insert the target peg in the new MDP although it converges the reward plateau slower. MPR-RL with hard-max weight also succeeds in learning the policy, but shows fluctuating performance. The other methods fail to learn the skill. It demonstrates that our approach can transfer knowledge from a family of MDPs to a novel MDP successfully. In both experiments we conclude that the adaptive weight predictor Ω among a family of MDPs is essential for transferring knowledge for our MPR-RL policy.

We evaluate the trained policies on hexagon peg-in-hole tasks with different goal positions and initial rotational angles between the peg and the workpiece. The success rates are shown in Table III. For all cases, MPR-RL with adaptive weight achieves highest success rates, although it has no prior knowledge of the new MDP. In comparison, when SPiRL has no access to prior knowledge of the current MDP, it shows poor insertion performance. Also, similar to the results shown in the maze navigation task, the prior predictor remarkably improves the performance of the trained policy.

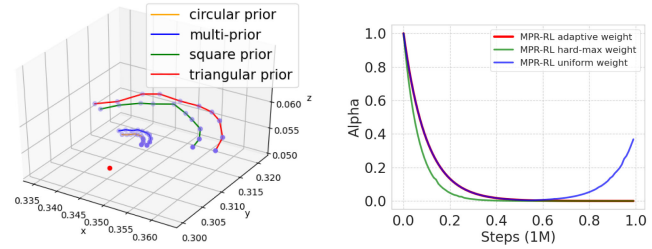
TABLE III
SUCCESS RATE FOR HEXAGON PEG-IN-HOLE TASK

Method	1cm	2cm	3cm	5°	10°	15°
MPR-RL adaptive	**0.86**	**0.80**	**0.82**	**0.90**	**0.85**	**0.72**
MPR-RL hard-max	0.65	0.62	0.53	0.72	0.63	0.58
MPR-RL uniform	0.42	0.32	0.18	0.39	0.22	0.20
SPiRL w/o knowledge	0.35	0.17	0.14	0.14	0.11	0.08



(a) (b)

Fig. 6. Impact of the number of priors and dataset size on performance of MPR-RL in maze navigation: (a) rewards for different numbers of priors; (b) rewards using different sizes of dataset.



(a) (b)

Fig. 7. (a) Comparing one action segment in hexagon peg-in-hole task. We generate trajectories of 10 waypoints using prior knowledge from different MDPs; (b) Comparison of tuning α through three weighting schemes, α cannot converge if we average all prior KL divergences (uniform weighting).

D. Ablation Evaluation

We investigate the impact of the number of priors and dataset size on the performance of our MPR-RL with adaptive weight method shown in Fig. 6. In the maze navigation task we observe that training with 3 or 5 priors yields similar performance, while 2 priors leads to diminishing reward. We suspect that it is hard to cover randomized dynamics of the maze environment with 2 priors. We also notice that the dataset size used for training each prior needs to be large enough to accelerate the learning of the policy. We conclude that our method needs enough prior knowledge from seen MDPs, but crucially not the target one, to benefit the learning the solution policy.

We compare one example of action sequences generated when using different skill priors in a hexagon peg-in-hole task shown in Fig. 7(a). Each trajectory is an action segment of 10 waypoints and the red point shows the position of the target hole. It can be seen that our MPR-RL method combines the knowledge from

three different MDPs circular, square and triangular and transfer the inserting skill to a new hexagon MDP. In this specific case, the action sequence from multi-prior is more similar to the circular one.

The regularization weight α in equation (8) is tuned automatically during training and the tuning procedure in the hexagon peg-in-hole experiment is shown in Fig. 7(b). As expected, α decreases because the sum of weighted prior divergences is getting close to the target divergence δ. It means that the regularization term of multiple priors accounts less and the skill policy learns to finish inserting task in the new MDP. We can see that α also converges by hard-max weighting scheme, but the fast converge by hard-max weighting restricts the exploration ability of the policy. In contrast, if the prior predictor is not used to compose multiple priors, the regularization term is not able to converge.

V. Conclusion and Future Work

We have presented an approach that learns multiple priors for a family of similar MDPs and compose these priors to guide the RL training of a policy on a new MDP. Our approach learns prior knowledge over specific skills for similar tasks. We evaluate our method on a simulated maze nagivation environment with different dynamics and a number of peg-in-hole task variants with a Franka Panda arm and demonstrate that our MPR-RL method can be adapted to the similar MDPs. By incorporating variable impedance into RL actions, we also show that our MPR-RL can be deployed directly on the real robot.

In our work, we utilize a neural network to predict the weight for each MDP prior that needs to be pre-trained and requires an amount of RL transitions. A promising future work might be to investigate how all priors can be composed more effectively (*i.e.* using Bayesian optimization). A second limitation is that our MPR-RL is only tested with a small family of MDPs. If a big family of MDPs are considered, it will be time and effort consuming to train these priors in advance. Therefore, it will be interesting to investigate how to learn a distribution over skill priors that can be adapted to a novel MDP. This could potentially mitigate the burden of collecting prior experience and accelerate transferring knwoledge between a big family of MDPs.

References

[1] K. Bousmalis *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 4243–4250.

[2] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 3803–3810.

[3] M. Hazara and V. Kyrki, "Transferring generalizable motor primitives from simulation to real world," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 2172–2179, Apr. 2019.

[4] Y. Du, O. Watkins, T. Darrell, P. Abbeel, and D. Pathak, "Auto-tuned sim-to-real transfer," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 1290–1296.

[5] W. Yuan, K. Hang, D. Kragic, M. Y. Wang, and J. A. Stork, "End-to-end nonprehensile rearrangement with deep reinforcement learning and simulation-to-reality transfer," *Robot. Auton. Syst.*, vol. 119, pp. 119–134, 2019.

[6] Q. Yang, A. Dürr, E. A. Topp, J. A. Stork, and T. Stoyanov, "Learning impedance actions for safe reinforcement learning in contact-rich tasks," in *Proc. NeurIPS Workshop Deployable Decis. Mak. Embodied Syst.*, Sydney, Australia, 2021.

[7] K. Pertsch, Y. Lee, and J. J. Lim, "Accelerating reinforcement learning with learned skill priors," in *Proc. Conf. Robot Learn.*, 2020, pp. 188–204.

[8] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," in *Proc. Conf. Robot Learn.*, 2017, pp. 334–343.

[9] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, no. 7, pp. 1633–1685, 2009.

[10] M. Wulfmeier, I. Posner, and P. Abbeel, "Mutual alignment transfer learning," in *Proc. Conf. Robot Learn.*, 2017, pp. 281–290.

[11] I. Arnekvist, D. Kragic, and J. A. Stork, "VPE: Variational policy embedding for transfer reinforcement learning," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 36–42.

[12] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 2169–2176.

[13] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," 2017, Accessed: *arXiv:1703.02949*.

[14] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 23–30.

[15] J. Tobin *et al.*, "Domain randomization and generative models for robotic grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 3482–3489.

[16] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Proc. Conf. Robot Learn.*, 2020, pp. 1162–1176.

[17] F. Muratore, F. Treede, M. Gienger, and J. Peters, "Domain randomization for simulation-based policy optimization with transferability assessment," in *Proc. Conf. Robot Learn.*, 2018, pp. 700–713.

[18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[19] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *Proc. Int. Conf. Learn. Representations*, 2019.

[20] X. Song, W. Gao, Y. Yang, K. Choromanski, A. Pacchiano, and Y. Tang, "ES-MAML: Simple hessian-free meta learning," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=S1exA2NtDB

[21] I. Clavera *et al.*, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2019.

[22] K. Arndt, M. Hazara, A. Ghadirzadeh, and V. Kyrki, "Meta reinforcement learning for sim-to-real domain adaptation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 2725–2731.

[23] G. Schoettler, A. Nair, J. A. Ojea, S. Levine, and E. Solowjow, "Meta-reinforcement learning for robotic industrial insertion tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 9728–9735.

[24] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5331–5340.

[25] K. Pertsch, Y. Lee, Y. Wu, and J. J. Lim, "Demonstration-guided reinforcement learning with learned skills," *Proc. 5th Conf. Robot Learn.*, 2021.

[26] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, and S. Levine, "Parrot: Data-driven behavioral priors for reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2020.

[27] Y. Zhu, P. Stone, and Y. Zhu, "Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4126–4133, Apr. 2022.

[28] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2014, Accessed: *arXiv:1312.6114*.

[29] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Int. Conf. Mach. Learn.*, 2014, pp. 1278–1286.

[30] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[31] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4RL: Datasets for deep data-driven reinforcement learning," 2020, Accessed: *arXiv:2004.07219*.