Kaushik, Rituraj; Arndt, Karol; Kyrki, Ville

**SafeAPT: Safe Simulation-to-Real Robot Learning Using Diverse Policies Learned in Simulation**

*Published in:*
IEEE Robotics and Automation Letters

*DOI:*
10.1109/LRA.2022.3177294

Published: 01/07/2022

*Document Version*
Publisher's PDF, also known as Version of record

*Published under the following license:*
CC BY

*Please cite the original version:*
Kaushik, R., Arndt, K., & Kyrki, V. (2022). SafeAPT: Safe Simulation-to-Real Robot Learning Using Diverse Policies Learned in Simulation. *IEEE Robotics and Automation Letters, 7*(3), 6838-6845. https://doi.org/10.1109/LRA.2022.3177294
SafeAPT: Safe Simulation-to-Real Robot Learning Using Diverse Policies Learned in Simulation
Rituraj Kaushik, Karol Arndt, and Ville Kyrki, Senior Member, IEEE

Abstract—The framework of sim-to-real learning, i.e., training policies in simulation and transferring them to real-world systems, is one of the most promising approaches towards data-efficient learning in robotics. However, due to the inevitable reality gap between the simulation and the real world, a policy learned in the simulation may not always generate a safe behaviour on the real robot. As a result, during policy adaptation in the real world, the robot may damage itself or cause harm to its surroundings. In this work, we introduce SafeAPT, a multi-goal robot learning algorithm that leverages a diverse repertoire of policies evolved in simulation and transfers the most promising safe policy to the real robot through episodic interaction. To achieve this, SafeAPT iteratively learns probabilistic reward and safety models from real-world observations using simulated experiences as priors. Then, it performs Bayesian optimization to select the best policy from the repertoire with the reward model, while maintaining the specified safety constraint using the safety model. SafeAPT allows a robot to adapt to a wide range of goals safely with the same repertoire of policies evolved in the simulation. We compare SafeAPT with several baselines, both in simulated and real robotic experiments, and show that SafeAPT finds high-performing policies within a few minutes of real-world operation while minimizing safety violations during the interactions.

Index Terms—Evolutionary robotics, learning from experience, machine learning for robot control.

I. INTRODUCTION

REINFORCEMENT learning (RL) is a promising direction towards allowing robots to acquire new skills through real-world interaction. Despite impressive results in simulated applications, such as simulated robots [1], the application of RL on physical systems is limited primarily due to the low data efficiency of these algorithms [2], [3].

In recent years, the idea of sim-to-real policy adaptation has become a promising alternative to improve data-efficiency in robot learning using RL [4], [5]. In this approach, a policy is learned first in simulation, and later adapted through real-world interactions to account for unmodeled or unknown variations between the simulation and the reality, often referred to as the reality gap.

To further improve the data efficiency in the sim-to-real policy adaptation approach, repertoire-based learning approaches can be used to optimize the policy on a discretized outcome-space, which is often of a lower dimensionality than the policy parameter space [6], [7]. The outcome-space can be defined as a user-defined space describing the outcome or the behaviour of the policies. For instance, for a robot hitting a hockey puck, the outcome-space can be defined as the 2D space of \( \langle x, y \rangle \) coordinate positions of the puck after executing the policy on the robot. Similarly, for a walking robot, the outcome-space can correspond to the different types of gaits produced by the policies. The core idea behind this approach is to evolve a large repertoire (i.e., a collection) of high-rewarding policies in simulation and associate each of them with a unique outcome in the discretized outcome-space. Then, on the physical robot, the optimal policy is chosen typically through Bayesian optimization [8] in the outcome-space. The main hypothesis of this approach is that, due to the diversity of the policies in the repertoire, some policies in the repertoire will still produce high rewards on the physical robot even in the presence of a large reality gap. For instance, a robot with a damaged leg can still walk if the repertoire has a policy to produce a walking gait that does not use the broken leg.

Nevertheless, due to the presence of the reality gap, there is no certainty that the policy learned in simulation is safe to be deployed on the physical robot. The execution of an unsafe...
policy may cause damage to the robot or its surroundings during the evaluation or adaptation.

In this work, we introduce Safety-Aware Policy Transfer (SafeAPT), a repertoire-based multi-goal learning approach. SafeAPT allows a robot to learn new skills in simulation and transfer them safely to the real world (Fig. 2). In the proposed approach, we first evolve a large repertoire of policies that achieve a diverse set of goals in simulation such that, for each goal, the associated policy performs the task as safely as possible within a distribution of diverse dynamics conditions. Then, on the physical robot, SafeAPT performs Bayesian optimization (BO) on the policy repertoire to maximize the reward for the specified goal while maintaining the desired safety constraint in each trial. To perform this constrained BO, we formulate an acquisition function, Expected safe improvement (ESI). ESI-BO uses two iteratively learned probabilistic models, corresponding to the reward and safety transformations. These models map the outcomes of the policies in the repertoire to the real-world rewards and safety scores. As each policy in the repertoire is associated with a unique outcome (i.e., the associated goal), the transformation models implicitly map the policies to their reward and safety score. To learn these models in a data-efficient manner, we incorporate the simulated results in the repertoire as priors for the models.

Our main hypothesis is that, due to the reality-gap, a policy repertoire evolved in simulation undergoes a transformation on the outcome-space for the real robot. As a result, the safety and reward associated with the policies are also transformed. We model these transformations with Gaussian process regression models [9] using the simulated results as prior mean functions.

We compare SafeAPT with three baselines, both in simulated and real-world experiments, and demonstrate that SafeAPT finds high-performance policies within a minute of real-world interaction while minimizing the safety constraint violations compared to the baselines.

II. RELATED WORK

Several prior works use probabilistic dynamical models to avoid unsafe behaviour during learning through trial-and-error [10]–[12]. For instance, in [12], the agent first uses a model-based RL approach to learn a probabilistic model to capture uncertainty about transition dynamics and catastrophic states. The model is then used in the real world for predicting and avoiding potentially unsafe states.

Shielding-based safe RL approaches typically use a safety-critic to estimate the safety of an action at the current state of the RL agent [13], [14]. If any action is predicted to be unsafe, the alternative safe action is executed by the agent. Typically, the Bellman equation is used to update the safety critic with sampled transitions from the current policy. However, while training the safety-critic, the agent may violate the safety constraints. Moreover, these approaches are not data-efficient enough to use on physical robots.

One class of optimization algorithms that has been successfully applied to robotics is Bayesian optimization (BO) [15], [16], particularly due to its ability to optimize black-box objectives which are expensive to evaluate. In [17], authors introduce a general framework to incorporate inequality constraints in Bayesian optimization. Similarly, [18], [19] propose safe Bayesian optimization in the context of parameter tuning in robotics. Nevertheless, BO does not scale well with the dimensionality of the parameters. Thus, on physical robots, BO is practically limited to optimizing around 10 parameters.

In order to scale up BO to high-dimensional parameter space, repertoire-based learning in robotics performs the policy optimization on the low dimensional outcome-space. The core idea behind this approach is to first pre-compute a large and diverse set of policies in simulation with a quality-diversity algorithm [20]–[22] and associate them with unique low-dimensional discrete outcomes. Then, an optimization process (such as BO) figures out the policy that works best in current dynamics conditions on that discrete outcome space [6], [23]–[25]. For instance, the IT&E approach [6] evolves a policy repertoire for a Hexapod robot to walk forward in simulation, but with different walking gaits (considering different gaits as the outcomes). On the physical robot with a high reality-gap due to a damaged leg, IT&E performs BO to figure out the gait (and so the associated policy in the repertoire) that makes the robot walk forward.

To incorporate safety into the IT&E approach, sIT&E [26] includes safety constraints as additional dimensions to the policy
repertoire. As such, the repertoire now contains diverse policies to perform the same task (e.g., to walk forward as fast as possible), with different behaviours or outcomes (such as different walking gaits) and with different safety scores. Given the safety constraints, sIT&E figures out the policy through trial-and-error using constrained BO [17]. The main limitation of sIT&E is that when the goal of the task changes in the real world, the repertoire needs to be evolved again, which is computationally expensive, typically taking several hours. In addition, as the number of safety constrained increases, so does the dimensionality of the repertoire, making BO prohibitively expensive.

To the best of our knowledge, none of the prior work on sim-to-real robot learning considers multi-goal and safe learning together. Unlike prior work, SafeAPT explicitly considers diverse dynamics conditions that the robot might face in the real-world while evolving the repertoire. In addition, SafeAPT is multi-goal and does not require the knowledge about the actual goal of the task a priori in simulation.

III. PROBLEM STATEMENT

We consider that the dynamics of the robot and the environment can be represented jointly with the following dynamical system:

\[ s_{t+1} = f(s_t, a_t, \psi) + w \]  

(1)

where \( f(\cdot, \cdot, \cdot) \) represents the state transition dynamics, \( s_t \) and \( a_t \) represent the state of the system and action applied at time \( t \), \( \psi \in \mathbb{R}^{d_\psi} \) is the dynamics parameter vector describing different dynamics conditions, and \( w \) is the i.i.d Gaussian noise accounting for any unmodeled dynamics and the inherent stochasticity of the system. We assume that the robot (our embodied agent) has access to \( f(\cdot, \cdot, \cdot) \) in the form of a physics simulator, but does not know the value of the dynamics parameter \( \psi_{\text{real}} \) in the real world. Instead, the robot has the knowledge about the distribution of feasible real-world dynamics parameters \( p(\psi_{\text{real}}) \).

The task has parametric goals \( g \in G \subset \mathbb{R}^{d_g} \). We assume that the robot is controlled by a deterministic policy (closed or open loop) \( \pi_\theta \) parameterized by \( \theta \in \mathbb{R}^{d_\theta} \) such that \( a_t = \pi_\theta(s_t, t) \). The execution of \( \pi_\theta \) on the system with dynamics parameters \( \psi \) results in the trajectory \( \tau = (s_0, a_0, s_1, a_1, \ldots, s_N) \) and the goal-space outcome \( g_\theta \). After execution of the policy, for any given goal \( g \), the robot receives a goal-space reward \( R(g_\theta, g) \) (which is inversely related to the distance between the goal-space outcome \( g_\theta \) and the specified goal \( g \)) and trajectory safety score \( C(\tau) \).

The robot has to solve the following optimization problem for a specified minimum safety score (or safety-limit) \( \lambda \) and goal \( g \) through episodic trial-and-error:

\[ \theta^* = \arg\max_{\theta} \mathbb{E}_{g_\theta \sim \pi_\theta} [R(g_\theta, g)] \]  

(2)

subject to \( \mathbb{E}_{\tau \sim \pi_\theta} [C(\tau)] \geq \lambda \).  

(3)

In addition, the constraint in (3) must be satisfied for any policy evaluation on the physical robot. In other words, we are not simply concerned about the safety of the optimal policy, but we want every policy evaluated on the robot during exploration to also be safe.

IV. APPROACH

SafeAPT consists of two main stages. We first generate a repertoire of policies that produce diverse goal-space outcomes in the simulation, while being as safe as possible on a distribution of simulated dynamics conditions of the real world. Then, to perform adaptation, we iteratively evaluate the most promising safe policies in the real world while using the collected data to train Gaussian process models describing the safety and reward in the real world, with data from simulation acting as a prior. We describe these stages in detail in the rest of this section.

A. Generating the Policy Repertoire in Simulation

The offline phase of SafeAPT starts by generating a policy repertoire. Our objective here is to obtain a large set of policies that are as safe as possible in simulation while covering the goal-space \( G \) of the task as widely as possible (i.e., each reachable discretized bin in \( G \) is assigned a policy that maximizes the safety score while reaching any goal in the given bin). A policy repertoire \( \Pi \) is a set of tuples \( (\theta, g_o, c) \), where \( \theta \) represents the policy parameters, \( g_o \in \mathbb{R}^{d_g} \) is the goal-space descriptor associated with the policy (e.g., the resulting Cartesian coordinate in a goal-reaching task), and \( c \) is the safety-score for the policy (the higher the better). It is to be noted here that SafeAPT does not assume any specific policy representation.

To simulate different dynamics conditions, we perform domain randomization by sampling \( N \) dynamics conditions \( \psi_{i=1:N} \) from \( p(\psi) \). For instance, a dynamics condition may describe the mass of the object that the robot is intended to manipulate, or the friction in the robot’s joints.

To generate the policy repertoire, we use the quality-diversity algorithm called MAP-Elites [20]. MAP-Elites first discretizes the goal-space \( G \) into \( K \) cells and randomly initializes \( M \) policies \( \theta_{i=1:M} \). Then, it evaluates these policies on each of the dynamics conditions \( \psi_{j=1:N} \) in the simulator, and creates the \( (\theta_i, g_o, c_i) \) tuples for \( i = 1:M \). Here, \( g_o \) is the mean goal-space descriptor and \( c_i \) is the minimum safety score obtained in all the dynamics conditions with the policy \( \theta_i \). Then, MAP-Elites attempts to insert the tuples into the respective cells in the repertoire based on their corresponding goal-space outcome. If two tuples fall in the same cell, the tuple with the maximum safety score is inserted. After this initialization, MAP-Elites performs the following steps iteratively until the policy evaluation budget is reached: (1) randomly pick a tuple \( (\theta_i, g_o, c_i) \) from the repertoire and add a small random variation to the policy \( \theta_i \), (2) evaluate the policy on all the dynamics conditions to create a new tuple, and (3) insert the new tuple into the repertoire if the cell is empty, or, replace an existing tuple by the new tuple having a higher safety score (discard the new tuple otherwise).

After repeatedly performing the above steps for a sufficient number of times, the repertoire will contain policies that are maximally safe in the simulation over the distribution of the feasible dynamics conditions.

B. Learning of the Reward and the Safety Model

In the real world, given a goal \( g \), we assign rewards to the tuples \( (\theta_i, g_o, c_i) \) in the repertoire using the goal-space reward function: \( r_i = R(g_o, g) \). These rewards are inserted into the respective tuples in the repertoire: \( (\theta_i, g_o, c_i, r_i) \).
We initialize two GP regression models that are used to learn a safety transformation function and reward transformation function in the goal-space \( T_c : G \rightarrow \mathbb{R} \) and \( T_r : G \rightarrow \mathbb{R} \). A GP model can be fully defined by the mean function \( \mathcal{M} (\cdot) \) and the covariance function \( k(\cdot, \cdot) \):

\[
T_c (\cdot) \sim \mathcal{G}P (\mathcal{M}_c (\cdot), k_c (g_0, g_0')) \quad (4)
\]

\[
T_r (\cdot) \sim \mathcal{G}P (\mathcal{M}_r (\cdot), k_r (g_0, g_0')) \quad (5)
\]

If \( D_{c,1} \) and \( D_{r,1} \) are the safety and reward observations in the real world for \( t \) policies from the repertoire, then the GPs can be calculated as:

\[
P(T_c (g_0)|D_{c,1}) = N (\mu_c (g_0), \sigma_c^2 (g_0)) \quad (6)
\]

\[
P(T_r (g_0)|D_{r,1}) = N (\mu_r (g_0), \sigma_r^2 (g_0)) \quad \text{where} \quad \mu_c (g_0) = \mathcal{M}_c (g_0) + k_c^T (K_c + \kappa \sigma_n^2 I)^{-1} (D_{c,1} - \mathcal{M}_c (g_0)) \]

\[
\mu_r (g_0) = \mathcal{M}_r (g_0) + k_r^T (K_r + \kappa \sigma_n^2 I)^{-1} (D_{r,1} - \mathcal{M}_r (g_0))
\]

\[
\sigma_c^2 (g_0) = k_c (g_0, g_0) - k_c^T (K_c + \kappa \sigma_n^2 I) k_c
\]

\[
\sigma_r^2 (g_0) = k_r (g_0, g_0) - k_r^T (K_r + \kappa \sigma_n^2 I) k_r
\]

\( \mathcal{M}_c (\cdot) \) and \( \mathcal{M}_r (\cdot) \) are prior mean-functions for safety and reward transformation models respectively. For any goal-space outcome \( g_{0i} \) in the repertoire, \( \mathcal{M}_c (g_{0i}) = c_i \) and \( \mathcal{M}_r (g_{0i}) = r_i \). \( \kappa \) and \( \sigma_n^2 \) are the prior noise for the GP models, \( K_c \) and \( K_r \) are the kernel matrices, \( k_c \) and \( k_r \) are the rows of the kernel matrices associated with the query \( g_0 \).

Equations (6) and (7) model how the safety and the reward are transformed in the real world compared to the values stored in the repertoire. For any policy \( \theta_i \) in the repertoire, the safety score and the reward can be predicted using the associated goal-space outcome \( g_{0i} \) using Equations (6) and (7).

C. Sim-to-Real Policy Transfer Using Bayesian Optimization

We modify the expected improvement (EI) acquisition function [8] of BO to filter out the policies in the repertoire that are potentially unsafe to execute on the robot. More concretely, we formulate an acquisition function called Expected Safe Improvement (ESI) as follows:

\[
ESI (g_0) = EI (g_0) \times 1_\lambda (g_0) \quad (8)
\]

where,

\[
1_\lambda (x) = \begin{cases} 0 & \text{if } LCB_c (g_0) < \lambda \\ 1 & \text{otherwise} \end{cases}
\]

(9)

\( LCB_c (g_0) \) is the lower confidence bound on the predicted safety for the policy corresponding to the goal-space outcome \( g_0 \) in the repertoire:

\[
LCB_c (g_0) = \mu_c (g_0) - \kappa \sigma_c (g_0), \kappa \in \mathbb{R}^+ \quad (10)
\]

In each episode, a new policy \( \theta^+ \) is selected from the repertoire by maximizing \( ESI (g_0) \):

\[
\theta^+ \Leftrightarrow g_0^+ = \arg\max_{g_0 \in \Pi} ESI (g_0) \quad (11)
\]

After each episode, the GP models are updated with the new observations (6) and (7). The process continues until the maximum number of trials is reached. Due to the clipping in the ESI function, SafeAPT only finds the maximum reward possible within the safety limit when the maximum reward is unsafe to attain.
D. Probability of Safety Violation

For any policy in the repertoire with associated goal-space outcome \( g_o \), the probability of the safety limit \( \kappa \) being violated can be computed using the Gaussian error function \( \text{erf}(\cdot) \) as

\[
Pr(c < \kappa) = \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{z}{\sqrt{2}} \right) \quad \text{where}, \quad (12)
\]

\[
z = \frac{\lambda - \mu_c(g_o)}{\sigma_c(g_o)} \quad \Rightarrow \lambda = \mu_c(g_o) + z\sigma_c(g_o) \quad \text{(13)}
\]

Now, using the \( ESI(\cdot) \) acquisition function (8) & (9), BO only considers policies with LCB on safety at least equal to \( \lambda \) for deployment on the real robot, i.e.,

\[
\lambda \leq \mu_c(g_o) - \kappa\sigma_c(g_o) \quad \text{(from (9) & (10))} \quad (15)
\]

Now using (14) in 15

\[
z \leq -\kappa \quad \text{(16)}
\]

Since \( \text{erf}(\cdot) \) is a monotonically non-decreasing function of \( z \), using 16 in 12:

\[
Pr(c < \kappa) \leq \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{-\kappa}{\sqrt{2}} \right) \quad (17)
\]

The inequality in 17 is the upper bound on the safety violation assuming that the GP accurately captures the mean and variance of the safety score associated with a policy in the repertoire. From 17 we see that a higher \( \kappa \) value lowers the upper bound on the probability of violating the safety limit. Intuitively, a higher \( \kappa \) value means that we are less certain about the mean prediction of the safety. Thus, BO selects policies that have mean safety prediction well above the specified safety limit, reducing the probability of failure. However, setting a very high value of \( \kappa \) will restrict BO from testing policies that are slightly risky but can potentially give a higher reward. In other words, \( \kappa \) controls the trade-off between the safety and reward maximization objectives.

V. EXPERIMENTAL SETUP

We evaluate SafeAPT on three simulated and one real-world setup, and compare the results with three baselines:

1) CBO: Constrained Bayesian Optimization with learned reward and safety models [17]; we expect this baseline to be less data-efficient as the optimization happens directly on the policy parameter space.

2) SafeAPT (no GP-safety): An ablation baseline of the proposed algorithm, where the sim-to-real safety transformation function is not learned from the real-world data; instead, the safety priors stored in the repertoire are assumed to be valid in the real world. This baseline shows the importance of learning the safety model from the real world interaction, even though the repertoire has policies that are potentially “at least safe” over a distribution of dynamics conditions.

3) SafeAPT (single dynamics): An ablation baseline of the proposed algorithm, where only one dynamics condition (randomly sampled from \( p(\psi) \)) is used to generate the policy repertoire. This baseline evaluates the importance of using multiple dynamics situations to generate the repertoire.

The goal of these experiments is to evaluate SafeAPT against the baselines in terms of (1) data-efficiency, (2) the rate of safety violations during the real-world trials, and (3) the performance (reward) of the final policy. All the simulated experiments are performed with several randomly chosen dynamics conditions at test time. We report the results for the worst case in terms of safety, i.e., the dynamics condition that has the highest number of safety violations over all the algorithms.

A. Asteroid Landing Task

In this task, a simulated asteroid lander has to find a policy that takes it to an altitude of 100 meters and hovers there. While learning the policy, the lander should not go below a safe altitude of 40 meters. For training, we sample 5 gravitational accelerations for each replicate from the uniform distribution \( U(3, 10) \text{ m/s}^2 \).

Here, the policy is a PID velocity controller, whose three coefficients as well as a sequence of five vertical velocity set-points collectively form the policy parameters \( 8D \text{ policy space} \). The duration of each episode is 15 seconds. The goal-space descriptor is the 1D altitude of the lander. The goal-space reward is inversely related to the distance between the desired altitude and the final altitude achieved after the execution of the policy on the lander. The safety score is the minimum altitude encountered in the trajectory.

B. Planar-Arm Goal Reaching Task

In this task, a 4-DoF planar kinematic arm (shown in the middle of Fig. 4) has to reach a specified 2D goal location (marked in blue) while avoiding 4 unsafe regions (marked in red). During learning, the end-effector should maintain at least 1 unit of distance from the unsafe regions. We use 10 combinations of link lengths for training – each link length is sampled from \( U(4, 7) \text{ units} \).

Here, the policy is a feed-forward neural network with 204 parameters that takes in the current joint angles as input and outputs joint velocity commands at every time-step. The episode length is 50 timesteps (5 seconds). The goal-space descriptor is the 2D location of the goal (scaled to \( [0, 1]^2 \)). The reward is inversely related to the distance to the goal from the final end-effector position. The safety score is defined as the closest distance to the unsafe regions from the end-effector encountered in the trajectory.

C. Kuka-Arm Hockey Task

This task involves a Kuka LWR 4+ robot arm hitting a hockey puck with a stick such that the puck slides to the desired target position, following [27], [28]. During learning, the puck should be at least 0.1 meters away from the edge of the table (the safety constraint). We sample 5 friction coefficient values from \( U(0.4, 0.7) \) during training. The simulation is done using MuJoCo [29]; the simulated setup is shown in Fig. 4.

In this setup, the policies are parametrized by \( \theta \in [-1, 1]^{119} \). To obtain joint position commands, \( \theta \) is passed to a denoising neural network model, pretrained on a wide range of striking motions — with different intensities and from different directions, resulting in a list of 17 joint setpoints, which are then interpolated using a cubic spline. The time required for each episode is 9 seconds. The goal-space descriptor here is the
Fig. 3. **GP updates in Asteroid landing experiment:** Plots show how reward and safety GP models are updated using the observations after each episode and how SafeAPT cautiously improves the reward while maintaining safety at each trial.

Fig. 4. **Simulated experiments.**

2D coordinate space on the table where the puck should land (scaled to $[0, 1]^2$). The reward is inversely related to the distance between the puck and the goal location. The safety score is defined as the distance from the puck to the closest edge of the table.

We also built a real-world version of the hockey-puck setup comprising of a Kuka LWR4+ arm (the same as was used in simulation) equipped with a plastic floorball stick. For the experiment, we used an ice hockey puck, with a whiteboard acting as a low-friction sliding surface. The position of the puck is measured by a ceiling-mounted Kinect camera. The safety area is demarcated by a row of wooden cubes, with the target position placed 10 cm away from the safety boundary. For this experiment, we use the same repertoires that were used in simulated experiments. The real-world setup is visualized in Fig. 1.

For all the experiments, we used squared exponential kernels [9] for the GP models. The hyperparameters of the GPs and BO are tuned through grid search in simulation by evaluating SafeAPT’s performance on “simulation-to-simulation” policy transfer with different dynamics conditions.

### VI. RESULTS AND DISCUSSION

To provide some intuition behind the adaptation process with SafeAPT, Fig. 3 shows the learning of the GP transformation models for the safety and the reward function in the Asteroid landing task. Thanks to the repertoire generated in the diverse simulated conditions, these GP models start with quite informative priors, which allow the models to learn with only a few data points from the real world. We observe that, in the first trial, the lander successfully hovers around 200 meters from the surface considering the high uncertainty about the safety of the policies below that level. The lander then cautiously tries policies from the repertoire that potentially improve the reward without violating the safety limit until it finds the policy that hovers the lander at the desired altitude of 100 meters.

From the plots for the simulated experiments (Fig. 5) we observe that SafeAPT finds at least as high rewarding policies as the baselines while maintaining the safety constraint throughout the whole adaptation process. On the other hand, the baselines violate the safety constraints during trial-and-error learning more frequently than SafeAPT (Table I). From Fig. 6 we also see that when the maximum rewarding policy is unsafe to execute, SafeAPT only obtains as high reward as possible within the safety limit.

Looking at the two ablations of SafeAPT, SafeAPT without the safety GP maximizes the reward greedily and is thus consistently violating the safety constraint in all the experiments. On the other hand, SafeAPT trained only on single dynamics is more effective at maintaining the safety constraint than SafeAPT without the safety GP (due to the learned safety model). However, it performs worse than complete SafeAPT due to the lack of diversity of dynamics in the simulations. As expected, due to performing the policy optimization directly on the high-dimensional parameter space, CBO is not able to compete with repertoire-based counterparts in terms of reward maximization. In terms of safety, CBO performs only slightly better than the other baselines.

In the sim-to-real Kuka hockey task, SafeAPT achieves not only a higher reward (out of the maximum possible reward of 1) but also complies with the safety constraint (no safety violations were observed in 8 replicates with independently generated repertoires; see Table II). Like in previous experiments, SafeAPT without the safety GP violates the safety constraint more frequently than the other baselines. Similar to the simulated setups, SafeAPT trained only on single dynamics shows fewer
Fig. 5. For the experiments (a)–(c) the plots A and B show the medians, 25 and 75 percentiles of the reward (plot A) and safety score (plot B) per episode for 15 replicates. Plot C shows the distribution of the executed policies on the reward-safety space.

To summarize, both the simulated and physical experiments confirm that, due to the lack of prior knowledge derived from diverse simulated situations, CBO fails in achieving satisfactory rewards and maintaining safety constraints. The ablation baselines SafeAPT (no GP safety) and SafeAPT (single dynamics) confirm that the diversity in the simulated conditions and learning of the safety transformation model help SafeAPT to not only achieve higher reward in a data-efficient manner.

Fig. 6. Hockey-puck experiment when the goal cannot be reached safely.

Safety violations due to the presence of the learned safety model. However, due to the lack of diverse dynamics conditions in simulation, it fails to achieve as good reward as SafeAPT in the real world. As expected, due to the high-dimensional policy parameter space, CBO fails to achieve as high rewards as the repertoire-based counterparts. Nevertheless, thanks to the constraint in BO, CBO violates the safety constraint less frequently than the other baselines.
but also to maintain the safety constraint during learning in the real world.

VII. CONCLUSION

Learning new skills through real-world interaction becomes challenging when, in addition to maximizing the reward, the robot must ensure safety during the interaction. In this paper, we introduced a sim-to-real multi-goal learning algorithm called SafeAPT for safe robot learning in the real world. SafeAPT inherits the typical limitation of repertoire-based learning, i.e., the pre-computed policies can be sub-optimal if the discretization of the goal space is not dense enough. Nevertheless, if further policy refinement is desired after performing SafeAPT, the fine-tuning of the policy can be performed on the parameter space safely using algorithms like Safe-Opt [19].

In SafeAPT, repertoire construction may take several hours. However, as it happens only once in simulation, it does not impact the performance of the algorithm in the real world. This process also scales well with the number of available CPU cores.

SafeAPT assumes that the reward function depends only on the goal-space outcome of a policy and the specified goal of the task. Thus SafeAPT may not scale well with complex reward functions. Nevertheless, we can still incorporate all the components of the reward function into the goal-space as long as MAP-Elites and BO do not get overwhelmed by the curse of dimensionality.

We believe that sim-to-real learning approaches like SafeAPT can be useful in goal-oriented robot-learning applications where a small mistake by the robot can incur a high cost or a complete failure of the mission.

ACKNOWLEDGMENT

We acknowledge the computational resources provided by the Aalto Science–IT project.

REFERENCES

[1] N. Heess et al., “Emergence of locomotion behaviours in rich environments,” 2017, arXiv:1707.02286.
[2] R. Kaushik, K. Chatzilygeroudis, and J.-B. Mouret, “Multi-objective model-based policy search for data-efficient learning with sparse rewards,” in Proc. Conf. Robot Learn., 2018, pp. 839–855.
[3] K. Chatzilygeroudis et al., “Black-box data-efficient policy search for robotics,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2017, pp. 51–58.
[4] R. Kaushik, “Data-efficient robot learning using priors from simulators,” Ph.D. dissertation, Université de Lorraine, Nancy, France, 2020.
[5] K. Boussmalis 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.
[6] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, “Robots that can adapt like animals,” Nature, vol. 521, no. 7553, pp. 503–507, 2015.
[7] R. Kaushik, P. Desreumaux, and J.-B. Mouret, “Adaptive prior selection for repertoire-based online adaptation in robotics,” Front. Robot. AI, vol. 6, p. 151, 2020, doi: 10.3389/frobt.2019.00151.
[8] E. Brochu, V. M. Cora, and N. De Freitas, “A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchial reinforcement learning,” 2010, arXiv:1012.2599.
[9] C. E. Rasmussen and C. K. Williams, Gaussian Processes for Machine Learning, vol. 2, no. 3, Cambridge, MA, USA: MIT Press, 2006.
[10] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, “A general safety framework for learning-based control in uncertain robotic systems,” IEEE Trans. Automat. Control, vol. 64, no. 7, pp. 2737–2752, Jul. 2019.
[11] L. Hewing, J. Kabzan, and M. N. Zeilinger, “Cautious model predictive control using gaussian process regression,” IEEE Trans. Control Syst. Technol., vol. 28, no. 6, pp. 2736–2743, Nov. 2020.
[12] J. Zhang, B. Cheung, C. Finn, S. Levine, and D. Jayaraman, “Cautious adaptation for reinforcement learning in safety-critical settings,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 11055–11065.
[13] M. Alishiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, “Safe reinforcement learning via shielding,” in Proc. Conf. Artif. Intell., 2018, pp. 2669–2678.
[14] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg, “Conservative safety critics for exploration,” in Proc. ICLR, 2021.
[15] B. Shahriari, K. Swersky, Z. Wang, R. F. Adams, and N. DeFreitas, “Taking the human out of the loop: A review of bayesian optimization,” Proc. IEEE, vol. 104, no. 1, pp. 148–175, Jan. 2015.
[16] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, “An experimental comparison of Bayesian optimization for bipedal locomotion,” in Proc. Int. Conf. Robot. Automat., 2014, pp. 1951–1958.
[17] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, “Bayesian optimization with inequality constraints,” in Proc. Int. Conf. Mach. Learn., 2014, pp. 937–945.
[18] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, “Safe exploration for optimization with gaussian processes,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 997–1005.
[19] F. Berkenkamp, A. Krause, and A. P. Schoellig, “Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics,” Mach. Learn., pp. 1–35, 2021. [Online]. Available: https://link.springer.com/article/10.1007/s10994-021-06019-1
[20] J.-B. Mouret and J. Clune, “Illuminating search spaces by mapping elites,” 2015, arXiv:1504.04909.
[21] A. Cully and Y. Demiris, “Quality and diversity optimization: A unified modular framework,” IEEE Trans. Evol. Comput., vol. 22, no. 2, pp. 245–259, Apr. 2018.
[22] K. Chatzilygeroudis, A. Cully, V. Vassiliades, and J.-B. Mouret, “Quality-diversity optimization: A novel branch of stochastic optimization,” in Black Box Optimization, Machine Learning, and No-Free Lunch Theorems. Springer, 2021, pp. 109–135.
[23] A. Cully and J.-B. Mouret, “Evolving a behavioral repertoire for a walking robot,” Evol. Computation, vol. 24, no. 1, pp. 59–88, 2016.
[24] M. Duarte, J. Gomes, S. M. Oliveira, and A. L. Christensen, “Evolution of repertoire-based control for robots with complex locomotor systems,” IEEE Trans. Evol. Comput., vol. 22, no. 2, pp. 314–328, Apr. 2018.
[25] A. Sharma et al., “Dynamics-aware unsupervised skill discovery,” 2019, arXiv:1907.01657.
[26] V. Papaspyros, K. Chatzilygeroudis, V. Vassiliades, and J.-B. Mouret, “Safety-aware robot damage recovery using constrained Bayesian optimization and simulated priors,” in Proc. BayesOpt’16 Workshop Conf. Neural Inf. Process. Syst., Barcelona, Spain, 2016.
[27] K. Arndt, A. Ghadirzadeh, M. Hazara, and V. Kyrki, “Few-shot model-based adaptation in noisy conditions,” IEEE Robot. Autom. Lett., vol. 6, no. 2, pp. 4193–4200, Apr. 2021.
[28] E. Todorov, T. Erez, and Y. Tassa, “MuJoCo: A physics engine for model-based control,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2012, pp. 5026–5033.