Learning Failure Prevention Skills for Safe Robot Manipulation

Abdullah Cihan Ak, *Graduate Student Member, IEEE*, Eren Erdal Aksoy, and Sanem Sariel, *Member, IEEE*

***Abstract*—Robots are more capable of achieving manipulation tasks for everyday activities than before. However, the safety of manipulation skills that robots employ is still an open problem. Considering all possible failures during skill learning increases the complexity of the process and restrains learning an optimal policy. Nonetheless, safety-focused modularity in the acquisition of skills has not been adequately addressed in previous works. For that purpose, we reformulate skills as base and failure prevention skills, where base skills aim at completing tasks and failure prevention skills aim at reducing the risk of failures to occur. Then, we propose a modular and hierarchical method for safe robot manipulation by augmenting base skills by learning failure prevention skills with reinforcement learning and forming a skill library to address different safety risks. Furthermore, a skill selection policy that considers estimated risks is used for the robot to select the best control policy for safe manipulation. Our experiments show that the proposed method achieves the given goal while ensuring safety by preventing failures. We also show that with the proposed method, skill learning is feasible and our safe manipulation tools can be transferred to the real environment.**

***Index Terms*—Robot Safety, Reinforcement Learning, Failure Prevention, Safe Robot Manipulation, Robust/Adaptive Control.**

I. Introduction

ROBOTS that are used in domestic environments require manipulation skills to perform various manipulation tasks [1]. Considering a kitchen environment, a robot can be assigned to cook various recipes such as soup or a cake. Primitive or compound motor skills such as stirring and pouring are needed to make these recipes. Existing learning methods enable learning such skills effectively [2], [3]. It is also crucial to ensure that these skills are executed safely. However, even well-designed skills are prone to fail in the real world due to wrong assumptions, perceptual errors, or changing environmental conditions [4]. These may threaten the integrity of the workspace. For example, a robot stirring soup may spill the soup or slide the pan out of the stove which can be harmful to the people nearby, its workspace, and itself. Therefore, the capability of the robot should not be limited to manipulation skills only, they also need to monitor/detect potential failures and prevent them for safety.

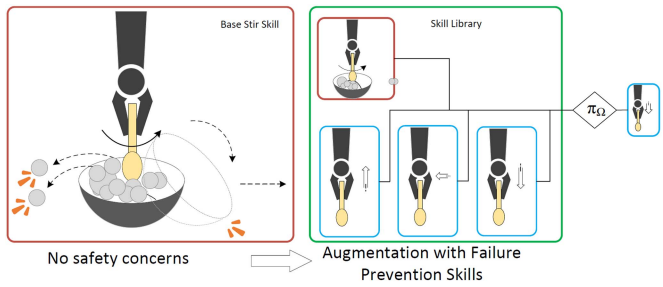


Fig. 1. A sample base skill, stir, may have been learned without taking safety concerns into account (spilling, sliding, or overturning). To make it safer, failure prevention skills are added to the skill library to be used when needed.

We propose a modular method where base skills are augmented by failure prevention skills, enhancing their safety. We group skills in the skill library into two categories according to their purposes: base skills and failure prevention skills. Skills with the purpose of reaching a goal to complete a task are defined as base skills (i.e., stirring or pouring). Skills with the purpose of preventing failures define failure prevention skills (such as prevention of sliding, overturning, or spilling). A skill selection creates a hierarchy and becomes the higher level, triggering the optimal skill from the skill library to safely accomplish the task. The lower level of the hierarchy is a skill library which is composed of base skills and failure prevention skills. For each potential failure, a failure detection and risk estimation model is defined to estimate the risk of the failure happening in the near future. Note that for the sake of simplicity, autonomous detection and identification of failures [4] is out of the scope of this work. Failure prevention skills are learned and added to the skill library to prevent potential failures. Fig. 1 illustrates this concept.

In the case of detecting a novel failure, the proposed method has the potential to learn a corresponding failure prevention skill. The skill library can easily be revised by only learning a novel failure prevention skill that mitigates this particular new failure type. Therefore, it becomes easier to adapt to new environmental conditions where novel failures could be encountered. While our work includes adapting to novel failures, discovering them is out of scope.

The proposed method is evaluated with a skill library formed in the simulated environment. Then it is transferred to a real

Manuscript received 30 April 2023; accepted 20 September 2023. Date of publication 13 October 2023; date of current version 24 October 2023. This letter was recommended for publication by Associate Editor Freek Stulp and Editor Jens Kober upon evaluation of the reviewers' comments. This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under Grant 119E-436. *(Corresponding author: Abdullah Cihan Ak.)*

Abdullah Cihan Ak and Sanem Sariel are with the Artificial Intelligence and Robotics Laboratory, Faculty of Computer and Informatics Engineering, Istanbul Technical University, 34469 Maslak-Istanbul, Türkiye (e-mail: akab@itu.edu.tr; sariel@itu.edu.tr).

Eren Erdal Aksoy is with the School of Information Technology, Center for Applied Intelligent Systems Research, Halmstad University, 30118 Halmstad, Sweden (e-mail: eren.aksoy@hh.se).

Digital Object Identifier 10.1109/LRA.2023.3324587

environment for real world evaluation. To the best of our knowledge, this is the first study that addresses learning reusable failure prevention skills to enable failure precautions into a skill library. Our main contributions are as follows:

- the formulation and implementation of a modular and hierarchical method for safe robot manipulation;
- enabling learning reusable failure prevention skills as precautionary switching policies in a skill library for safe manipulation;
- adapting to failures and augmenting incrementally;
- real world applicability by effective safety precautions in the physical world.

II. Related Work

Manipulation skills are often considered closed-loop sensorimotor control systems for robots to complete various tasks. With recent developments in reinforcement learning (RL), even though robots can learn and use manipulation skills effectively for a limited number of objectives, they struggle when the number of objectives increases due to the curse of dimensionality. Therefore, RL-based skill learning rather focuses on achieving a given task without considering potential failures. Safe reinforcement learning approaches [5], [6], [7], [8] apply learning to satisfy the goal while keeping the execution safe by estimating the risk. These approaches have the intention to prevent failures by learning to avoid unsafe states, however, they do not respond to scenarios with several failures as in unstructured environments. A previous work [9] addresses this issue but with a limited number of failures.

Representing a skill as smaller skills combined with a hierarchy [10], [11], [12] benefits from reduced complexity of the goal for optimal skill learning. While such skills can be predefined, it is also possible to learn smaller skills incrementally [13], [14]. Even though hierarchical approaches focus on learning several skills, none of the learned skills have an explicit focus on ensuring the safety of the execution. For that purpose, the safety and the goal can be considered as two different objectives, and learning safe and robust skills can be expressed as a multi-objective reinforcement learning problem by decomposing the reward. Separating rewards and punishments can effectively perform both the task and failure prevention (*contextual inhibition* and *spontaneous recovery*) [15]. In the context of multi-objective reinforcement learning, positive rewards (reward) can be used to learn how to complete the task (main objective), whereas negative rewards (risk) can be introduced to avoid potential failures (safety objective). Many recent works [16], [17], [18], [19], [20] decompose the reward to learn to achieve the goal safely, but they only respond to a single failure type. It can make a crucial difference to balance both rewards and risks, whereas several objectives would result in suboptimal results for each objective. Especially, failures and their consequences are not limited and can vary in a dynamic scene, therefore, an increased number of objectives can make the multi-objective reinforcement learning problem unfeasible.

We present a modular and hierarchical method for safe robot manipulation in a multi-objective setting. Different from previous works explained above, our method includes an explicit intention to ensure the safety of the manipulation against several failures using failure prevention skills. By decoupling the learning of base and failure prevention skills, our method allows robots to incrementally add failure prevention skills without altering the base skill policies. We also show that learned individual failure prevention skills can further be transferred to relatively different tasks.

III. Learning Failure Prevention Skills for Safe Robot Manipulation

Cognitive robots are equipped with either hand-coded or learned motor skills to achieve a given task. Even though these skills work well for controlled environments, in unstructured environments, their outcomes are not always as expected, and even worse, undesired/unsafe situations may occur due to failures in changing situations. To ensure safety, it is crucial for robots to anticipate potential failures before they occur, and to prevent them if possible. In this work, we address this problem and formulate it as augmenting base robot skills with appropriate precautions to make them safer without changing them.

We define a base skill, π_b, as a motor skill to achieve either a primitive or a compound action (e.g., *pick and place objects, pour liquid, stir a bowl of ingredients*, etc.). During performing this skill in a setting different from the trained one, failures are inevitable due to either wrong assumptions, incorrect estimations or unanticipated situations [4]. Let ρ_k be a risk representing such a failure probability (between 0 and 1) for failure k. The safety of the execution of π_b needs to be monitored by continually checking the occurrence of a finite set of risks: $\{\rho_0, \ldots \rho_n\}$ which are modeled in the design time or discovered online. The main problem that we address in this study asks; when the execution of a base skill (π_b) increases the risk of a failure (ρ_k), to learn and activate necessary motor skills that enable a transition to a safe state. We call this type of complementary skill, a failure prevention skill π_{p_k} that is responsible for reducing a corresponding risk. Note that, these failure prevention skills are generic and reusable motor skills that can be used to augment different base skills. Therefore, the problem asks for learning these failure prevention skills ($\pi_{p_0}, \ldots, \pi_{p_n}$) and learning when these skills should take over the control to ensure the safety of the whole execution. Thus, a dynamic chain of skills (i.e., a linear sequence of the base skill and the failure prevention skills selected at different occurrences) is executed based on the circumstances of the world. This problem also asks for an effective and efficient selection of failure prevention skills during execution.

The skill library L of the robot includes both the base skills and failure prevention skills (1). L can be gradually built by adding novel skills online. Each learned skill extends the skill library to make it robust against each failure type that is observed.

$$L = \{\pi_{b_0}, \ldots, \pi_{b_m}\} \cup \{\pi_{p_0}, \ldots, \pi_{p_n}\} \tag{1}$$

In this work, we mainly focus on the augmentation of a base skill (π_b) with the necessary safety precautions that are learned through experimentation by the robot. It is crucial to showcase

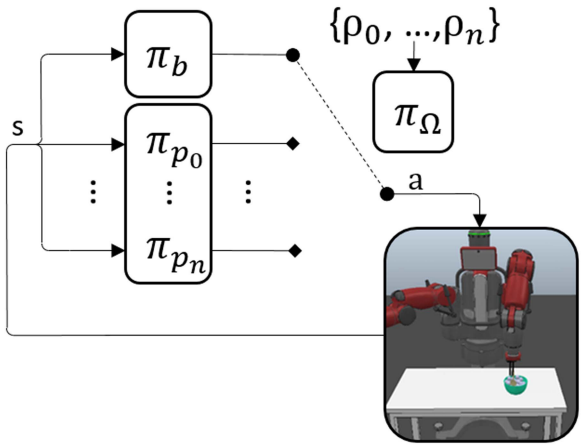


Fig. 2. Hierarchical Failure Prevention Model. A corresponding failure prevention skill is triggered if the state matches a risk model.

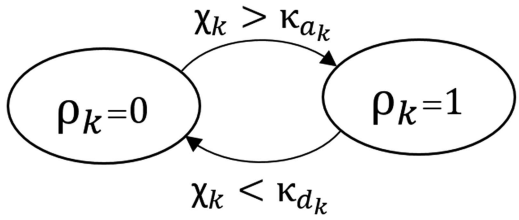


Fig. 3. Safe ($\rho_k = 0$) - Risky ($\rho_k = 1$) transition model considering the thresholds.

that a chain of skills selected from the library based on observations can be effective in ensuring safety without changing the base skill. Therefore, in our current formulation, risk estimation models are determined preliminary; and the discovery of novel failures is left as a near future work. However, novel failure discovery is discussed in Section III-B. We propose a hierarchical failure prevention model to solve this problem which is depicted in Fig. 2. This model learns failure prevention skills (π_{p_k}) themselves, and uses a a skill selection policy (π_Ω) to activate a corresponding one from L during the execution of a base skill (π_b) to reduce a given risk (ρ_k). The following subsections describe these two-level learning processes in detail.

A. Learning a Base Skill

A base skill (π_b) can be hand-coded by an expert or learned by optimizing models such as Markov Decision Process (MDP) and Dynamic Movement Primitives (DMP) [21] using reinforcement learning (RL) or learning from demonstration (LfD). However, these models may not work as well as desired when the robot is exposed to reality, as unexpected failures are more likely in different settings other than the trained one. Therefore, a learned skill needs to be adapted to changes that occur in the environment. However, this adaptation may degrade the effectiveness of actual task performance (i.e., base skill) for the sake of adapting to changes. Therefore, we propose to augment the base skill to make it safer without changing it. This also ensures a more reusable solution to be used as a library of skills that can be used to augment other base skills as well.

B. Failure Detection and Risk Estimation

A risk (ρ_k) is a binary safety estimate (*safe* - *risky*) against a failure, and presents the probability of a failure to occur in the near future. In order to continue the execution safely, failure prevention decisions should be given based on risk estimations in real time.

Risk models can be predefined, or risks may be discovered online. Online discovery of failures necessitates a lifelong learning procedure for detecting and classifying novel situations. This can be done by using a (semi-)supervised learner outputting probabilities for known cases, and detecting anomalies by checking the probability of the assigned class label or using an unsupervised approach to isolate anomaly cases [22], [23]. Another way of online discovery is to synthetically learn to create failure cases on nominal execution. We have recently proposed such an approach using adversarial RL to both create failure cases and learn to prevent them [24]. Our preliminary results show that this method has the potential to discover failures online. However, we leave a detailed analysis of such a method as future work. In this letter, we primarily focus on learning generic failure prevention skills and the integration of these motor skills to a base skill.

Thus, we use a rule-based risk estimation model which is expressed as a finite state machine $\rho_k = FSM(\chi_k, \kappa_{a_k}, \kappa_{d_k})$ as illustrated in Fig. 3. For failure k, a risk ρ_k uses an observable parameter χ_k from the environment, and evaluates the risk using the activation threshold κ_{a_k} and the deactivation threshold κ_{d_k} where $\kappa_{a_k} \neq \kappa_{d_k}$. A safe state is transitioned to a risky state if χ_k is observed to be larger than κ_{a_k}. A risky state is transitioned to a safe state if χ_k is observed to be smaller than κ_{d_k}. The interval between κ_{a_k} and κ_{d_k} prevents undesired fluctuations between states when the observed parameter is close to thresholds.

C. Learning Failure Prevention Skills

In our method, a failure prevention skill is modelled with MDP and learned to prevent a risky situation. The reward of a failure prevention skill is determined by the corresponding risk estimation model as given in (2). Since risk estimations are binary, the reward is sparse. During the training, the failure prevention skill policy is optimized to maximize the reward as it minimizes the risk.

$$R_k = 1 - \rho_k \tag{2}$$

Risky states are expected to be observed occasionally. To learn to prevent risky states, the robot should observe these cases quite often. However, this is not the case if it is not intended. In order to improve the sample efficiency of learning a failure prevention skill, the robot should be able to experience risky states more often. For that purpose, specific procedures are designed to create different failure situations.

D. Skill Selection

Once the skill library L is formed, the robot can execute the task safely with a skill selection policy (π_Ω) that arbitrates overall skills in L. π_Ω achieves this selection in a hierarchical fashion (See Fig. 2). π_Ω selects a skill from L using risk estimations $\{\rho_0 \ldots \rho_n\}$, thus, it is expected to select an appropriate failure prevention skill to reduce a risk to prevent any catastrophic situations before they occur, and select the base skill when appropriate to complete the task.

We use a rule-based mapping from risk estimations to skills in L to form the skill selection policy π_Ω. With π_Ω, the robot executes the base skill π_b in a state without any risk. With the observation of risk ρ_k, the corresponding failure prevention skill π_{p_k} is selected to reduce the risk ρ_k. Upon detection of multiple risks, predefined priorities between failure prevention skills determine which skill to execute first. Priorities are determined by the impact of the failure on the manipulation's safety. Note that the risks are sorted from most important to least important, therefore, the failure prevention skill corresponding to the risk with the greatest index is selected first.

IV. Empirical Evaluation of Augmenting Stir Skill

In this section, we present our case study on a continuous *stirring* task. To accomplish this task, a Baxter humanoid robot uses a spoon to stir particles in a bowl. The goal is for the robot to move particles in the bowl for a given time. During the nominal execution of the *stirring* skill, one natural failure is observed: *spilling* the particles from the bowl. For different objects, environments, and/or control conditions two additional failures might occur in our setup, *sliding* the bowl and *overturning* the bowl. Failure prevention skills are learned and used to augment the *stir* skill for the safety of the execution.

In the real world environment, a Baxter humanoid robot with a spoon attached to its gripper is situated in front of a table. A bowl ($r = 8$ cm, $h = 8$ cm) is placed on the table, and several sphere-shaped particles ($r = 1 \pm 0.5$ cm) are placed in the bowl. In front of the table, a Kinect One RGBD sensor is placed to track the bowl and the particles. The real environment is shown in Fig. 4(a). The simulated environment is created using CoppeliaSim and designed similarly to the real environment. 40 sphere-shaped particles are placed in the bowl. The number of the particles is selected according to the competence to observe the mentioned failures. During the simulated experiments, red and blue colored particles are used to observe the performance on the *stirring* task by the human eye. The simulated environment is shown in Fig. 4(b).

In the simulated environment, two setups are used to obtain optimal skill policies; the fixed bowl setup (F) and the unrestricted setup (U). The fixed bowl setup restricts the effect of forces the robot can apply on the bowl, resulting in the bowl keeping its position and orientation. Therefore, the robot focuses on the goal without considering failures related to the bowl pose. Then, the unrestricted setup is used for learning failure prevention skills for taking into account the previously ignored failures. Forces acting on the bowl result in failures such as sliding and overturning.

(a) (b)

Fig. 4. Experimental setups. (a) Real (b) Simulated Environment.

First, we present our method in the simulated environment. Details of learning the stir base skill, forming risk estimation models, and learning failure prevention skills are given in Section IV-A, Section IV-B, and Section IV-C respectively. The skill selection among these skills and the evaluation of the proposed method that uses these skills is given in Section IV-D. Then, the learned skills are transferred into the real environment, and the proposed method is evaluated in the real world; whose results are presented in Section IV-E. The additional materials and videos of both simulated and real robot are available[1].

A. Learning Stir Base Skill

We use the *stir* skill as the base skill in our study. With the *stir* base skill, the robot uses a spoon to continuously stir the particles inside a bowl without considering any potential failures. Therefore the fundamental goal of the *stir* base skill is to move particles as much as possible. By focusing on only one objective while omitting potential failures, we reduce the problem from multi-objective learning to single-objective learning, making the learning more straightforward.

In this work, we formulate the skill learning problem as an MDP with a tuple: $\langle S, A, T, R, \gamma \rangle$ where $s_t \in S$ is a continuous state, $a_t \in A$ is a continuous action, $T(s_{t+1}|s_t, a_t)$ is transition probability, $R(s_t, a_t, s_{t+1})$ is the reward and γ is the discount factor. In continuous state/action environments, Deep Deterministic Policy Gradients (DDPG) [25] is one of the most prominent approaches for skill learning. With DDPG, a skill is denoted by two policies; an actor policy π and a critic policy Q. The actor policy π maps states $s_t \in S$ to actions $a_t \in A$ in a continuous domain, and the critic policy Q estimates values of state-action $s_t - a_t$ pairs. For exploration, a noise ν from an Ornstein-Uhlenbeck [26] process is used. Acting according to a_t results in a transition to state s_{t+1}, thus, the robot obtains a reward $r_t \in R$ depending on the task to learn. Transitions (s_t, a_t, s_{t+1}, r_t) are collected into an experience replay memory,

[1][Online]. Available: https://air.cs.itu.edu.tr/projects/tubitak-119e436.html

and they are used for the optimization of the target actor and the target critic policies. Actor and critic policies are updated with target policies periodically.

The *stir* skill policy π_b is learned as the base policy in the simulated environment. The policy has been optimized by formulating the execution as an MDP with the following state and action spaces:

$$S_{stir} = [x_{spoon}, \phi]$$
$$A = [\Delta x_{spoon}] \tag{3}$$

where S_{stir} is composed of the position of the spoon relative to the bowl (x_{spoon}) and the phase(ϕ) of the execution, and A is composed of the displacement of the position of the spoon (Δx_{spoon}). Parameters in the state and the action spaces are selected by their relevance to the problem and for simplicity, positions are represented with 2D coordinates of the plane parallel to the table. x is the Cartesian positions between $[-\eta, \eta]$ where η is the safety perimeter for the robot to operate within the allowed range. ϕ is the phase value representing the relative time of the execution, making the system time-variant. Using ϕ adds a temporal aspect to the skill, and we experimentally found it useful for periodic movements such as *stir* skill. It is a value between $[0, \phi_{\max}]$, and updated after each movement decision with ϕ_{step} using:

$$\phi = (\phi + \phi_{step}) \; mod \; \phi_{\max} \tag{4}$$

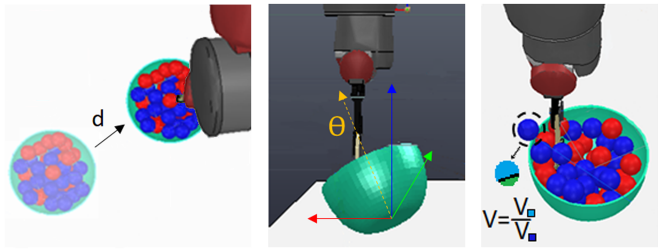
Stir base skill is expected to reflect the essence of stirring without safety concerns, reducing the complexity of the problem. Therefore, an optimal *stir* base skill policy that focuses specifically on stirring is obtained with a fixed bowl setup. Stirring can be considered as the continuous movement of the particles in the bowl hence the reward is formulated as the sum of the displacement of the particles in 50 ms. The reward function is given as follows:

$$R_b(s_t, a_t) = \sum_{k=0}^{n} r_b(x_t^k, x_{t+1}^k),$$
$$r_b(x_t^k, x_{t+1}^k) = \begin{cases} ||x_t^k - x_{t+1}^k||_2, & \text{if } \; in(x_{t+1}^k, bowl) \\ 0, & \text{else} \end{cases} \tag{5}$$

where r_b is the displacement of a particle between two consecutive states 50 ms apart if the particle is in the bowl and 0 otherwise. n is the number of particles in the bowl which is 40 in our setup. R_b is the reward of the base skill which is the sum of individual rewards for each particle.

Both actor and critic neural networks are designed with two linear feed-forward layers with 400 and 300 neurons, respectively and with ReLU activation layer in between. The networks are trained with Deep Deterministic Policy Gradients (DDPG) [25] for 1500 episodes with 500 steps where the batch size is 128, learning rates(α_a, α_c) are 0.0001 and discount factor (γ) is 0.99. For exploration, linearly decaying epsilon is used and the noise is modeled with Ornstein-Uhlenbeck [26] process with parameters $\mu_\nu = 0$, $\sigma_\nu = 1$, $\theta_\nu = 0.15$.

As the result of the training, the best policy π_b is selected as the optimal policy for the *stir* base skill and added to L.



(a) (b) (c)

Fig. 5. Observed failures during testing the learned *stir* base skill. (a) Sliding the bowl (b) Overturning the bowl (c) Spilling contents from the bowl.

TABLE I
PARAMETERS THAT ARE USED FOR RISK ESTIMATION MODELS

Failure Type	χ	κ_a	κ_d
slide	d	0.05m	0.02m
overturn	θ	0.3rad	0.1rad
spill	V	0.66	0.33

B. Risk Estimation Models

Potential failures that may occur are observed during test executions of π_b in the simulated environment. When the *stir* base skill (π_b) is tested in the fixed bowl setup, *spill* failure is observed. *spill* failure happens when the particles in the bowl get out of the bowl due to forces applied by the spoon. However, when the *stir* base skill (π_b) is tested in the unrestricted setup, two additional failures are observed: *slide* and *overturn* failures. During the execution, the bowl is expected to stay close to the starting position. But forces applied on the spoon may slide the bowl away from the starting point resulting in *slide* failure. Especially, the increased amount of particles results in the bowl sliding away frequently. While the bowl slides away, the friction between the bowl and the table can act as a hinge, rotating the bowl and spilling most of the particles, resulting *overturn* failure. Depictions of these failures are shown in Fig. 5. These failures are used as testbeds to learn failure prevention skills to augment the stir skill for safety.

Relevant risk estimation models are designed as finite state machines(FSM) with two states, *safe* and *risky*. An observed parameter greater than risk activation κ_a triggers the risk. Risk disappears when the observed parameter is reduced to risk deactivation κ_d. Designed risk estimation models for *stir* base skill (π_b) and their empirically selected parameters are given in Table I.

The risk estimation model of *slide* failure uses the distance (d) between the initial location of the bowl (x_0), and the current location (x_t) of the bowl, calculated as $d = \Delta(x_0, x_t)$. The risk estimation model of *overturn* failure uses the rotation angle (θ) between the initial pose of the bowl (θ_0) and the current pose (θ_t) of the bowl, calculated as $\theta = \theta_t - \theta_0$. A spill risk can be estimated by the volume of the particles that are out of the bowl. The ratio of the overflow of each particle (V_n) is calculated as $(V_{e_n} - (V_{e_n} \cap V_b))$ where V_{e_n} is the volume of the particle and V_b is the volume of the bowl. Then the overflown volume(V) is calculated as the maximum ratio of overflow ($max_n(V_n)$) among the particles. The pose and the volume of the bowl and

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

particles are directly acquired from the simulation as its ground truth value.

C. Learning Failure Prevention Skills

Failure prevention skills are acquired by learning failure prevention policies for corresponding risk estimation models. They are learned in the simulated environment with the same network design and optimization parameters as that of learning the base skill (Section IV-A). Different from base skill learning, these skills have different state representations and reward functions. Additionally, they use initial procedures to increase the risk at the start of the episode letting the robot encounter the corresponding risk easily.

Failure prevention skills require additional parameters related to the failure they respond to on top of the base skill state representation. They use d, θ, and V parameters from the corresponding risk estimation model to learn the optimal robot movement that avoids the risk. The state of a failure prevention skill is obtained by concatenating S_{stir} with $\chi = \{d, \Theta, V\}$ from the corresponding risk estimation model; d for *slide* failure, θ for *overturn* failure and V for *spill* failure as given in (6). The reward for each failure prevention skill is sparse and acquired with the corresponding risk estimation model as explained in (2).

$$\begin{aligned} S_{slide} &= [x_{spoon}, \phi, d] \\ S_{overturn} &= [x_{spoon}, \phi, \theta] \\ S_{spill} &= [x_{spoon}, \phi, V] \end{aligned} \tag{6}$$

To learn a failure prevention skill effectively, risky states should be experienced during training which may not occur often enough. Initial procedures are designed to move the robot to a risky state, thus, letting the robot experience risky states. Each failure has different initial procedures. For prevention of *slide*, the initial position of the bowl is sampled randomly triggering the risk ρ_{slide}. For prevention of *overturn*, the robot moves the spoon toward a random direction until the risk $\rho_{overturn}$ is triggered. For prevention of *spill*, the robot moves the spoon toward a random location in the bowl until the risk ρ_{spill} is triggered. The episode starts once the corresponding risk is triggered using initial procedures, and the robot learns how to reduce the risk.

As the result of trainings, the best policies of $\pi_{p_{slide}}$, $\pi_{p_{overturn}}$, $\pi_{p_{spill}}$ are selected as optimal failure prevention policies for sliding, overturning, spilling failures, respectively. The selected policies are added to the skill library forming L_4.

D. Overall Results

A skill library is initialized with the learned base skill π_{stir} (Section IV-A). Then, learned failure prevention skills (Section IV-C) are added to the skill library (L) augmenting the base skill for safe robot manipulation. We define safe robot manipulation as skill selection in real time from the skill library consisting of a base skill to complete the task and failure prevention skills to reduce failure risks. We use a rule-based skill selection policy (Section III-D). The priorities are given in (7) where $I(\pi_{p_k})$ is the importance of failure k which is determined by its effect on the task.

$$I(\pi_{p_{overturn}}) > I(\pi_{p_{spill}}) > I(\pi_{p_{slide}}) \tag{7}$$

Note that, for a different setup, the importance of failures can be different from what we present. For example, if the robot stirs a pan on a stove, keeping the pan on the stove would be more important than spilling the content.

1) Evaluation of the Augmented Stir Skill: The comparative results are presented in Table II where all methods are tested for 20 episodes with 1000 steps. The table reports the mean and the standard deviation of performance measures (R, d, θ, V). We first evaluate π_b in the fixed bowl setup for benchmarking (π_b-*F*). Evaluation results indicate that π_b stirs particles effectively while spilling occasionally. Then, our method is evaluated in the unrestricted setup (L_4-*U*). Comparing π_b-*F* and L_4-*U*, we can claim that the proposed method is significantly better for failure prevention with a tradeoff of stir efficiency. The loss of stir efficiency is tolerable as the environment of our method is more challenging and vulnerable to failures than the former. Therefore, it uses its time effectively to prevent any of the failures and stir whenever it is safe.

For a fair comparison between our method and π_b, the latter is also tested in the unrestricted setup (π_b-*U*). Comparing π_b-*U* and L_4-*U*, we see that our method is safer with a tradeoff of stir efficiency. Note that even though the number of spill events decreased for π_b-*U*, one would not conclude that π_b-*U* is safer than π_b-*F*. Because in the unrestricted setup, forces affecting particles get diminished since a part of the force is transferred to the bowl, causing the number of spill events to decrease. Note that no overturn event is detected in the results because overturn failure occurs when the robot interacts with the bowl which only happens with $\pi_{p_{slide}}$. While this never happens for π_b-*F* and π_b-*U*; L_4-*U* successfully prevents overturn failure with $\pi_{p_{overturn}}$.

2) Adaptability to Novel Failures: To show the adaptability of our method, we show how a robot working in the fixed bowl setup adapts its library to the unrestricted setup. In the fixed bowl setup, only spill failure can be observed since the bowl is fixed, and the skill library L_2 is formed with π_b and $\pi_{p_{spill}}$. When restrictions on the bowl are removed from the environment, novel failures; *sliding* and *overturning* are observed, and $\pi_{p_{slide}}$ and $\pi_{p_{overturning}}$ skills are learned to prevent them, respectively. Now the skill library is extended (L_4) with these skills. When we compare L_2-*F* and L_4-*U*, it can be seen that with our method, it is easy to adapt to new conditions by discovering novel failures and learning corresponding failure prevention skills.

3) Modularity vs Compound Skill: One of the main questions that should be discussed is whether modularity helps with the failure prevention problem or not. For this investigation, a compound base-failure prevention skill(π_c) is learned that takes into account all three failures during learning to stir and penalizes accordingly. π_c is trained in the unrestricted setup with the same training setup (Section IV-A), except for the reward function. As the reward, the sum of all rewards for each objective is used

TABLE II
EVALUATION RESULTS OF AUGMENTATION OF THE STIR SKILL WITH FAILURE PREVENTION SKILLS IN THE SIMULATED ENVIRONMENT

	Stir Reward ↑		Spill ↓		Slide ↓		Overturn ↓	
	mean	std	mean	std	mean	std	mean	std
π_b-F	329.00	17.57	**4.55**	1.50	N/A	N/A	N/A	N/A
π_b-U	251.29	12.90	**2.20**	1.32	**0.11**	0.01	0.00	0.00
L_4-U	159.64	34.67	0.20	0.52	0.05	0.03	0.00	0.00
L_2-F	87.20	53.98	0.10	0.30	N/A	N/A	N/A	N/A
π_c-U	126.76	10.70	0.15	0.36	0.02	0.01	0.00	0.00

The total displacement of the particles in the bowl in meters (The Stir Reward), the number of the occurred spill events (Spill), the average position difference of the bowl from the safe location in meters (Slide), and the number of the occurred overturn events (Overturn) are reported. The upward arrow (↑) indicates that the higher is the better, and the downward arrow (↓) indicates that the lower is the better. F indicates fixed bowl setup and U indicates unrestricted setup. Bold indicates undesired outcomes.

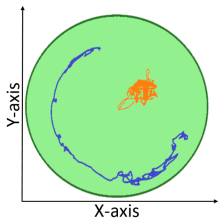


Fig. 6. Trajectory of a randomly selected particle from the bowl travels using the compound skill π_c-U(orange) and the modular method L_4-U(blue).

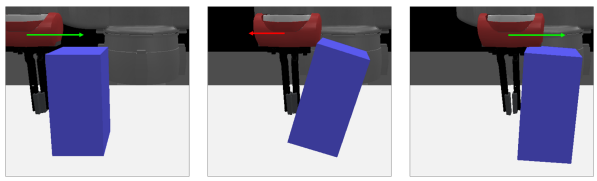


Fig. 7. An example execution trace of pushing (base skill) a rectangular prism: (a) The robot pushes the prism to the right (b) While pushing, the prism starts to overturn, and the robot decides to execute $\pi_{p_{overturn}}$ (c) After executing the failure prevention skills, the robot continues to execute the base skill.

as given in (8).

$$R_c = R_b + R_{slide} + R_{overturn} + R_{spill} \tag{8}$$

Comparing L_4-U and π_c-U from Table II, we can deduce that our method performs slightly better for the stirring efficiency, and the compound skill performs slightly better for failure prevention. However, when we compare the learned stir patterns of both methods (see a randomly selected particle's trajectories in Fig. 6), we see that π_c-U does not perform a circular movement. It rather moves the spoon linearly in a narrow area resulting in only a slight change of particle locations which is not an effective stir. This performance degradation also supports the decrease in the average stir reward. Due to this linear pattern of movement, the probability of failures is smaller compared to L_4-U. On the other hand, our further analysis shows that the modular method's reward is highly dependent on how fast the prevention policy reduces the risk which explains the high standard deviation of the stir reward of L_4-U and L_2-F.

4) Reusability of Failure Prevention Skills Over Different Base Skills: To show the reusability of learned failure prevention skills, we set a case scenario on a different base skill, *push*, where an overturning failure may occur. The same $\pi_{p_{overturn}}$ skill which was previously learned for preventing an overturning failure for *stir* is included in the library. We have observed that by using the same risk estimation model, the robot can detect the risk of overturning and prevent the failure by using this skill. An example execution trace is given in Fig. 7. Note that $\pi_{p_{overturn}}$ skill can successfully augment different actions (stir and push in our case). However, some skills may not be reusable and specific to the base skill or the manipulated object. For example, for $\pi_{p_{slide}}$, stir pushes the bowl from the inside to fix the position. This skill is reusable for container type objects but it may not be useful for other types. The robot may need to experiment with the utilities of failure prevention skills for different situations. We leave a broader analysis of the reusability of prevention skills as future work.

E. Transfer to the Real World

In this work, we directly transfer d and θ parameters from the simulation to the real world. However, we use domain adaptation for the rest of the parameters(V) that can not be represented directly.

For the *stir* base skill, the position of the spoon x_{spoon} which is attached to the gripper is obtained by the kinematic chain of the robot. ϕ is initialized as 0 and $\phi_{\max}$ is set to 50. For $\pi_{p_{slide}}, \pi_{p_{overturn}}, \pi_{p_{spill}}$, the pose of the bowl is observed using a particle filter-based tracking algorithm [27]. Then d and θ are estimated from the pose of the bowl. Detecting and tracking particles in the bowl is impractical in the real world. In order to obtain V, we use a point cloud filtering approach using Point Cloud Library (PCL), sampling points from particles to calculate the highest point on the z-axis ($max_n(z_n)$).

By transferring the L_4 model, we have shown our method's capability to prevent failures in the physical world. Since the exact positions of the particles in the bowl cannot be observed directly as in the simulation, the cumulative reward of the execution can not be determined in the real world. Therefore, the model's success can be stated qualitatively. Based on our observations, we can conclude that continuous *stirring* with preventative skills

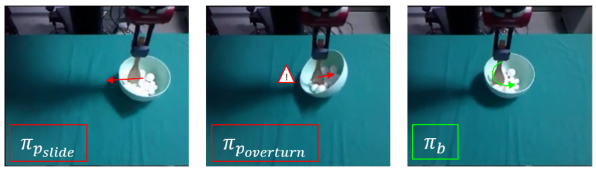


Fig. 8. Example execution trace of L_4 model in the real world: (a) The robot uses $\pi_{p_{slide}}$ to correct the position of the bowl (b) While executing $\pi_{p_{slide}}$, the bowl starts to overturn and the robot executes $\pi_{p_{overturn}}$ (c) Then, the robot continues to *stir* with π_b.

is successfully applied. An example execution trace is given in Fig. 8.

We conducted additional tests to analyze the performance of L_4 model's failure prevention capability. At first, L_4 model is tested for each failure individually by starting the scenario in a risky condition. In these tests, the robot easily avoided failure and continued with the base skill after the risk is reduced. Then L_4 model is tested for 2-minute-long executions. In these tests, different amounts of particles are used. When the bowl is about 70% or less full, the robot runs as expected, it prevents failures in risky states, and stirs continuously in safe states. When the bowl is more than 70% full, the robot can not reduce the $max_n(z_n)$ easily and gets stuck in failure prevention. In some cases with a large number of particles, the actions of some particles may pop out of the bowl. This failure is not observable by the available risk estimation models, yet may be estimated by using additional sensors such as force sensors which are not available in this work. Additionally, L_4 model is tested against human interventions; changing the location of the bowl, adding additional particles, and removing particles from the bowl. The model adapts robustly to new conditions by preventing possible failures without stopping which are available in our website.

V. Conclusion and Future Work

In this work, we propose a modular method where base skills and failure prevention skills are combined. Failure prevention skills are learned for preventing potential failures and used for augmentation of base skills to make them safer by using rule-based skill selection. Evaluation results indicate that the learned failure prevention skills help base skills to complete their tasks safely. The method is also extendable upon novel failures. We also transferred learned skills to be used in the real world successfully. Additional materials, and videos of simulated and real results are available online. In the near future, we plan to extend our approach to show how the skill library is incrementally built with failure prevention skills for novel failures discovered online. A broader analysis of the reusability of failure prevention skills for more complex scenarios is also related to this extension and is on our agenda.

References

[1] M. Ersen, E. Oztop, and S. Sariel, "Cognition-enabled robot manipulation in human environments: Requirements, recent work, and open problems," *IEEE Robot. Automat. Mag.*, vol. 24, no. 3, pp. 108–122, Sep. 2017.

[2] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[3] D. Kalashnikov et al., "QT-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," 2018, *arXiv:1806.10293*.

[4] A. Inceoglu, E. E. Aksoy, A. C. Ak, and S. Sariel, "FINO-net: A deep multimodal sensor fusion framework for manipulation failure detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 6841–6847.

[5] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg, "Conservative safety critics for exploration," in *Proc. Int. Conf. Learn. Representations*, 2021.

[6] S. Cheong, J. Lee, and C. Kim, "A new concept of safety affordance map for robots object manipulation," in *Proc. IEEE 27th Int. Symp. Robot Hum. Interactive Commun.*, 2018, pp. 565–570.

[7] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 4310–4319.

[8] B. Eysenbach, S. Gu, J. Ibarz, and S. Levine, "Leave no trace: Learning to reset for safe and autonomous reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2018.

[9] A. C. Ak, A. Inceoglu, and S. Sariel, "When to stop for safe manipulation in unstructured environments?," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 1767–1769.

[10] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *J. Mach. Learn. Res.*, vol. 22, pp. 1395–1476, 2021.

[11] O. Kroemer, C. Daniel, G. Neumann, H. van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 1503–1510.

[12] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, 2017.

[13] A. Bagaria and G. Konidaris, "Option discovery using deep skill chaining," in *Proc. 37th Int. Conf. Learn. Representations*, 2020, pp. 1–21.

[14] A. Bagaria, J. Senthil, M. Slivinski, and G. Konidaris, "Robustly learning composable options in deep reinforcement learning," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, 2021, pp. 2161–2169.

[15] R. Lowe and T. Ziemke, "Exploring the relationship of reward and punishment in reinforcement learning," in *Proc. IEEE Symp. Adaptive Dyn. Program. Reinforcement Learn.*, 2013, pp. 140–147.

[16] H. Van Seijen, M. Fatemi, J. Romoff, R. Laroche, T. Barnes, and J. Tsang, "Hybrid reward architecture for reinforcement learning," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5398–5408.

[17] Z. Lin, L. Zhao, D. Yang, T. Qin, T.-Y. Liu, and G. Yang, "Distributional reward decomposition for reinforcement learning," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 6215–6224.

[18] Z. Lin, D. Yang, L. Zhao, T. Qin, G. Yang, and T. Liu, "RD^2: Reward decomposition with representation disentanglement," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 11298–11308.

[19] S. Elfwing and B. Seymour, "Parallel reward and punishment control in humans and robots: Safe reinforcement learning using the MaxPain algorithm," in *Proc. IEEE 7th Joint Int. Conf. Develop. Learn. Epigenetic Robot.*, 2018, pp. 140–147.

[20] J. Wang, S. Elfwing, and E. Uchibe, "Deep reinforcement learning by parallelizing reward and punishment using the maxpain architecture," in *Proc. IEEE 8th Joint Int. Conf. Develop. Learn. Epigenetic Robot.*, 2018, pp. 175–180.

[21] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models formotor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, 2013.

[22] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.

[23] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019, *arXiv:1901.03407*.

[24] M. C. Kutay, A. C. Ak, and S. Sariel, "Adversarial learning of failure prevention policies," in *Proc. IEEE 31th Signal Process. Commun. Appl. Conf.*, 2023, pp. 1–4.

[25] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[26] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Phys. Rev.*, vol. 36, no. 5, 1930, Art. no. 823.

[27] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3195–3202.