Safe Manipulation via Task-Relevant Reach-Avoid Reinforcement Learning

Saumya Saxena
Robotics Institute
Carnegie Mellon University
Email: saumyas@andrew.cmu.edu

Andrea Bajcsy
Robotics Institute
Carnegie Mellon University
Email: abajcsy@andrew.cmu.edu

Oliver Kroemer
Robotics Institute
Carnegie Mellon University
Email: okroemer@andrew.cmu.edu

***Abstract*—Reach-avoid reinforcement learning is a promising approach for computing low-level control policies for nonlinear dynamical systems with complex safety and goal specifications. However, leveraging this framework for manipulation tasks presents a fundamental challenge: the need to reason about safety with respect to *all objects* in the environment. In cluttered scenes, defining the state to include all objects is computationally intractable; moreover, many objects induce different constraints depending on their semantic properties (e.g., soft vs. fragile objects), making scalable specification of the safety constraints challenging. Our key insight is that only a subset of entities in the scene are task- and safety-relevant during interactive manipulation. We therefore propose parameterizing our safe control policies with a significantly smaller subset of task- and safety-relevant entities, which we infer online during interaction. Offline, we pre-train a parameterized safety policy by considering diverse combinations of task objectives, safety-relevant objects, and safety constraints. Online, we explore a suite of methods—from hand-designed heuristics to vision-language models (VLM)—to dynamically update the task- and safety-relevant entities over the deployment horizon. We evaluate our approach in simulation, wherein a Franka Panda manipulator must dynamically, but carefully, pull a target object from a stack, while accounting for nearby clutter, and safely place it into a goal region. We discover a tradeoff between the robot's lower-dimensional state space, environment complexity, and the overall ability to safely succeed at the task. Furthermore, we find that the semantic priors within VLMs can enable these models to select relevant objects—and their corresponding constraint types—more effectively than hand-designed heuristics, enabling safer and more performant policies.**

I. Introduction

For complex manipulation tasks in cluttered environments, the full state space and the comprehensive set of task and safety constraints can be prohibitively large, rendering policy learning both intractable and inefficient. However, for a specific task, only a subset of objects and constraints, determined by the task description and the metric and semantic properties of the environment, are typically relevant, while others can be ignored without compromising performance or safety. For example, when sliding a book out from beneath a stack of books, the relevant objects may include the stack itself and any fragile objects nearby, such as a cup of coffee that must be avoided. In contrast, interactions with soft objects, like toys, can be safely permitted. Furthermore, as the task progresses and the book is moved along the table, the set of relevant objects may evolve dynamically. It is crucial for the policy to adapt accordingly to account for these changes.

Figure 1: Dynamic manipulation task: The robot dynamically, but carefully, slides a target object from under another object, and moves it to a goal region while avoiding safety-critical objects on a cluttered table.

This work aims to learn safe policies for interactive and dynamic manipulation tasks in cluttered environments. Hamilton-Jacobi Reachability (HJR) analysis is a prominent method for ensuring safety for dynamical systems that guarantees constraint satisfaction by finding an optimal solution to a reach-avoid problem. However, it suffers from scalability issues as the dimension of the state space increases. Recent works [14, 3, 16], take inspiration from reinforcement learning-based approaches to find approximate solutions to the reach-avoid problem in higher dimensions by utilizing a time-discounted reach-avoid bellman update (DRABE [16]) with contraction mapping properties. Despite its promise to scale, leveraging this framework for dynamic and interactive manipulation tasks is quite challenging due to combinatorial increase in the number of *interactions and contact types* that need to be considered as the environment becomes more complex. Additionally, most works focus on collision avoidance and do not consider a *safe dynamic interaction* problem, which is a key consideration in dynamic manipulation tasks. We train a policy that considers dynamic interactions between a subset of task-relevant and

safety-critical objects and constraints, significantly reducing the state space considered during safety analysis.

In this work, we propose an approach for safely performing dynamic and interactive manipulation tasks using parameterized safe policies that condition on a subset of dynamically changing relevant objects and corresponding task and safety constraints, by inferring them online during execution. Offline, we train a parameterized safe policy, using reach-avoid reinforcement learning (RARL), in a reduced state space, considering diverse combinations of task and safety constraints. Online, we explore heuristic-based and vision-language-based methods to infer relevant objects and constraints. We show that while heuristic-based methods such as k-nearest neighbor aid in identifying relevant objects based on their metric properties, Vision-Language models (VLMs) are proficient in identifying objects and constraints based on semantic properties. We perform experiments in simulation on a challenging task of dynamically, but carefully, pulling a target object from under a stack and moving it to a goal region while avoiding safety-critical objects on a cluttered table. Our results demonstrate that DRABE policies outperform vanilla RL approaches and have fewer dynamic safety violations. Furthermore, we observe that task- and safety-relevant parameterized safety policies perform comparable to or outperform policies trained with privileged full-state information.

The key contributions of this work are as follows:

- A scalable approach for learning safe policies for dynamic and interactive manipulation tasks in clutter using reach-avoid reinforcement learning.
- An offline training pipeline for learning a parameterized policy that considers a reduced set of objects and diverse combinations of task and safety constraints.
- Online, we explore heuristic-based and vision-language based methods for identifying relevant objects and constraints based on both metric and semantic properties.
- Through results in simulation on a challenging dynamic and interactive manipulation task, we demonstrate that task- and safety-relevant dimensionality reduction enables scalable and efficient policy synthesis while maintaining safety guarantees, paving the way for deploying safe manipulation policies in unstructured environments.

II. Related work

Reachability-based Safe Control Synthesis. Hamilton-Jacobi reachability analysis [5, 21, 23] provides theoretical formulations for finding the solution to nonlinear reach-avoid control problems by minimizing the worst case (minimum over time) loss. Compared to approaches that minimize the cumulative loss over time [7, 4], HJR analysis provides rigorous safety assurances, ensuring that the system avoids unsafe states under worst-case scenarios. However, HJR becomes computationally intractable as the dimension of the state space increases [10]. Recent methods [14, 3, 16] take inspiration from RL-based approaches [27, 31] to extend reachability analysis to higher dimensions using a discounted formulation of the reach-avoid bellman equation. Leveraging HJR analysis for learning safe policies in dynamic, cluttered manipulation remains underexplored due to the high dimensionality of complex multi-body interactions. Our key contribution is reducing this complexity by focusing only on task- and safety-relevant constraints, identified through both semantic and metric properties.

Safe Control for Robotic Manipulation. Impedance and null space control are widely used to ensure compliant interactions in robotic manipulation. Impedance control modulates stiffness and damping properties to enable safe and adaptive interactions with the environment [15, 1], while null space control allows secondary objectives like collision avoidance to be incorporated without interfering with primary tasks [26, 11]. In contrast, contact-aware controllers explicitly reason about contact interactions to keep interaction forces below a safety threshold [18, 36]. Recent work has focused on learning contact-aware complaint controllers using expert demonstrations [25, 2] and RL [34, 35, 22]. Our work builds on these efforts by enabling contact-aware control through reachability-based methods in cluttered environments.

Semantic-aware Safe Planning using VLMs. Recent works have explored incorporating semantic safety into task planning. [29] integrate safety prompts in the code-as-policies [19] setup, while [32] use LLMs to decompose high-level tasks into subtasks and verify them against LTL specifications. Other approaches infer user preferences through demonstrations and active queries [30]. Semantic constraints derived from VLMs have been used for planning, with simulators verifying feasibility [17, 9]. However, these approaches primarily focus on quasi-static tasks without long-horizon reasoning. Most relevant to our work is [8], which incorporates dynamic constraints within a formal safety framework. However, it requires manually designing barrier functions for each constraint type, and the policy accounts for all constraints simultaneously, making it impractical for cluttered environments. Our approach leverages RARL to learn safety value functions that can be solved for both *liveness* and *safety* based on a subset of relevant task and safety constraints.

III. Background

A. Hamilton-Jacobi Reachability Analysis

Hamilton-Jacobi Reachability (HJR) analysis provides a formal approach for computing safe policies that guarantee constraint satisfaction by finding an optimal solution to a reach-avoid problem. Consider a discrete-time dynamical system $s_{t+1} = f(s_t, a_t)$, where t is the current time step, $s_t \in \mathcal{S} \subset \mathbb{R}^n$, $a \in \mathcal{U} \subset \mathbb{R}^m$, $\mathcal{U}$ is compact and dynamics f are bounded and Lipschitz continuous. Target set $\mathcal{T} \subset \mathcal{S}$ describes *reach* states that satisfy $\{s : l(s) > 0\}$ and failure set $\mathcal{F} \subset \mathcal{S}$ describes *avoid* states that satisfy $\{s : g(s) < 0\}$. $\mathcal{T}$ and $\mathcal{F}$ are closed sets and $l(s), g(s) : s \to \mathbb{R}$ are Lipschitz continuous functions. The safety problem is to find the *reach-avoid* set, $\mathcal{RA}(\mathcal{T}; \mathcal{F})$, which is the set of states from which a controller can drive the system to $\mathcal{T}$ while avoiding $\mathcal{F}$ at all times t. It was demonstrated in [14] that states belonging to the

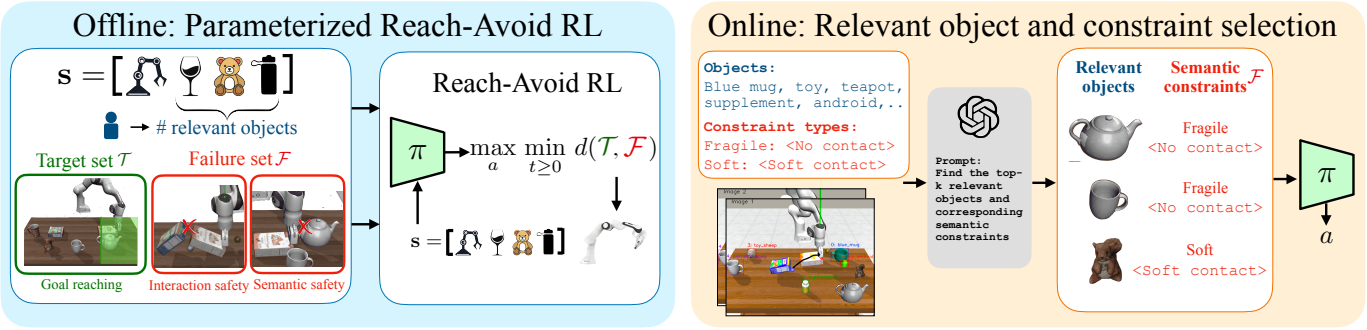

Figure 2: Method overview: Offline (left), we train a parameterized policy using reach-avoid reinforcement learning for a user-specified number of relevant objects and a diverse combination of task and safety constraints. Online (right), the relevant objects and task and safety constraints are inferred using vision-language-based or other heuristic-based methods, and used by the pretrained policy to execute the next action.

set $\mathcal{RA}(\mathcal{T};\mathcal{F})$ satisfy $V(s) > 0$, where $V(s)$ is the solution to the following fixed-point Reach-Avoid Bellman equation (RABE):

$$V(s) = \min\left[g(s), \max\left(l(s), \sup_{a\in\mathcal{U}} V(f(s,a))\right)\right] \quad (1)$$
$$V(s) > 0 \Leftrightarrow s \in \mathcal{RA}(\mathcal{T};\mathcal{F})$$

It is important to note that, unlike the Bellman updates in dynamic programming based approaches [6], (1) does not have a time-discounting term, thus it does not induce a contraction mapping and cannot converge to a fixed point using value iteration.

B. Reach-Avoid Reinforcement Learning (RARL)

Based on time-discounting in temporal difference learning [27] and Q-learning [31] based methods, [14] introduced a time-discounting term in RABE (1), thus inducing contraction mapping in value function learning, which was extended to the reach-avoid setting and deep Q-learning by [16]. The Discounted Reach-Avoid Bellman Equation (DRABE) [16] can be written as:

$$V(s) = \gamma \min\left[g(s), \max\left(l(s), \sup_{a\in\mathcal{U}} V(f(s,a))\right)\right] + (1-\gamma)\min\left(l(s), g(s)\right) \quad (2)$$

where γ is the discount factor, which can be interpreted as the probability of episode continuation.

IV. Method

An overview of our proposed approach is shown in Fig. 2. We consider the task of dynamic and interactive manipulation in cluttered tabletop environments. In this section, we first describe the problem setup in Sec. IV-A, our approach for training parameterized safe policies in Sec. IV-B and finally discuss different methods for online identification of relevant objects and constraints in Sec. IV-C.

A. Problem setup

For a robot manipulation task in a tabletop cluttered environment as shown in Fig. 1, the *full* state of the system is represented as $\mathbf{s} = [\mathbf{s}^{\text{EE}}, \mathbf{s}^o_1, \ldots, \mathbf{s}^o_N] \in \mathcal{S}$ where $\mathcal{S}$ is the full state space. $\mathbf{s}^{\text{EE}} = [\mathbf{x}^{\text{EE}}, \dot{\mathbf{x}}^{\text{EE}}]$ is the state of the end-effector where $\mathbf{x}^{\text{EE}} \in \mathbb{R}^3$ is the cartesian position and $\dot{\mathbf{x}}^{\text{EE}} \in \mathbb{R}^3$ is the cartesian velocity. The state of each object is represented as $\mathbf{s}^o_i = [\mathbf{x}^o_i, \dot{\mathbf{x}}^o_i], i \in \{1,\ldots,N\}$ where N is the total number of objects on the table. Robot actions are represented as $\mathbf{a}^{\text{EE}} = \Delta\mathbf{x}^{\text{EE}} \in \mathbb{R}^3$, the cartesian displacement of the end-effector. Target set $\mathcal{T} = \{\mathbf{s} : l(\mathbf{s}) > 0\}$ is the set of all states that satisfy all task objectives such that $l(\mathbf{s}) = \min(l_1(\mathbf{s}), \ldots, l_p(\mathbf{s}))$ where p is the number of task objectives. Failure set $\mathcal{F} = \{\mathbf{s} : g(\mathbf{s}) < 0\}$ is the set of all states that violate any of the failure constraints such that $g(\mathbf{s}) = \min(g_1(\mathbf{s}), \ldots, g_q(\mathbf{s}))$ where q is the number of safety constraints. For a given task, at a certain time step, a subset of objects and constraints might be relevant. The relevant state space is represented as $\mathbf{s}_{\text{rel}} = [\mathbf{s}^{\text{EE}}, \mathbf{s}^{o_{\text{rel}}}_1, \ldots, \mathbf{s}^{o_{\text{rel}}}_{N_{\text{rel}}}]$ where N_{rel} is the number of relevant objects, the relevant target set $\mathcal{T}_{\text{rel}} = \{\mathbf{s}_{\text{rel}} : l^{\text{rel}}(\mathbf{s}_{\text{rel}}) > 0\}$ where $l^{\text{rel}}(\mathbf{s}_{\text{rel}}) = \min(l^{\text{rel}}_1(\mathbf{s}_{\text{rel}}), \ldots, l^{\text{rel}}_{p_{\text{rel}}}(\mathbf{s}_{\text{rel}}))$ and the relevant failure set $\mathcal{F}_{\text{rel}} = \{\mathbf{s}_{\text{rel}} : g^{\text{rel}}(\mathbf{s}_{\text{rel}}) < 0\}$ where $g^{\text{rel}}(\mathbf{s}_{\text{rel}}) = \min(g^{\text{rel}}_1(\mathbf{s}_{\text{rel}}), \ldots, g^{\text{rel}}_{q_{\text{rel}}}(\mathbf{s}_{\text{rel}}))$. p_{rel} and q_{rel} are the number of relevant task and safety constraints, respectively.

B. Offline Parameterized Safe Policy Learning

We utilize Deep Deterministic Policy Gradient (DDPG) [20] as our reinforcement learning framework to learn a parameterized safe policy in continuous action spaces. The Q-function update is modified to the discounted reach-avoid formulation (2). Suppose Q_ϕ is the Q-network, μ_θ the policy network, and $Q_{\phi_{\text{targ}}}$ and $\mu_{\theta_{\text{targ}}}$ are the target Q-network and target policy networks respectively. Given a sample (s, a, r, s', d) from the replay buffer $\mathcal{D}$, the Mean-squared Bellman Equation (MSBE) in DDPG is as follows:

$$L(\phi,\mathcal{D}) = \mathop{\text{E}}_{(s,a,r,s',d)\sim\mathcal{D}}\Bigg[\Big(Q_\phi(s,a) - \big(r + \gamma(1-d)Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))\big)\Big)^2\Bigg]$$

where $r = \min(l(s), g(s))$ is the reward and d indicates whether state s' is terminal. The Discounted Reach-Avoid

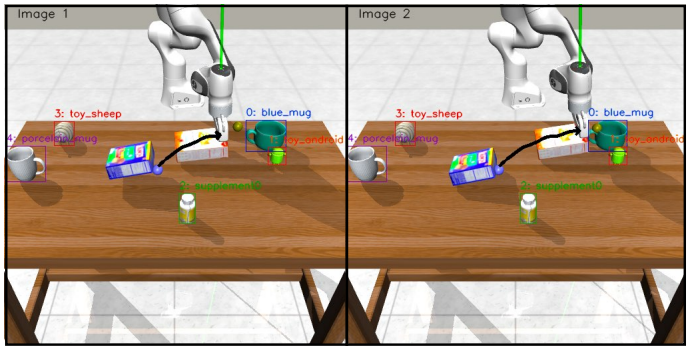

Figure 3: Sequence of images input to Dynamic-VLM for relevant object selection. The images are annotated with object bounding boxes as well as the past trajectory (black arrow).

Bellman Error for the DDPG algorithm can be written as:

$$L(\phi, \mathcal{D})_{\text{DRABE}} = \mathop{\mathrm{E}}_{(s,a,r,s',d)\sim\mathcal{D}}\Bigg[Q_\phi(s,a) - \Bigg[(1-\gamma)\min(l(s), g(s)) + \gamma \min\Big\{g(s), \max\{l(s), Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))\}\Big\}\Bigg]\Bigg]$$

In cluttered environments, the number of relevant objects for a given task is typically much smaller than the total number of objects in the scene, $N_{\text{rel}} \ll N$. We start by choosing an appropriate value for $N_{\text{rel}} = k$ based on task requirements. Let the state space that encompasses k objects and the robot be represented as $\mathcal{S}_k$ and let $\mathcal{T}_k$ and $\mathcal{F}_k$ represent the corresponding sets of all possible distinct task and safety constraints. For a certain subset of objects and their semantic properties, a different set of constraints might be relevant. For example, consider the scenario where $k = 1$ and $N = 2$, where there are two objects (a toy and a coffee cup) on the table, and one is selected at a time. The relevant safety constraints are g_1 (soft contact for the toy) and g_2 (collision avoidance for the coffee cup). Since the relevant object is unknown at runtime, we train a parameterized policy that adapts to both constraints based on the selected object. During training, for each episode, the initial state of the system is sampled from $\mathcal{S}_k$ and all combinations of constraints are sampled from $\mathcal{T}_k \times \mathcal{F}_k$. A policy trained in $\mathcal{S}_k$ is denoted as $\pi^{\theta}_{N=k}$ where θ is the parameterization of the policy with constraints from the set $\mathcal{T}_k \times \mathcal{F}_k$.

C. Online Relevant Object and Constraint Inference

For a given task, we perform relevant object and constraint selection based on two types of object attributes: metric and semantic. Metric attributes include kinematic and dynamic object properties, such as position, velocity, and mass. Semantic attributes encompass intrinsic and contextual properties, such as material, color, texture, and affordance. To synthesize performant as well as safe policies, it is essential to consider both sets of attributes for the interacting objects.

Online Relevant Object Selection: We explore three techniques for selecting relevant objects in a scene: Static-kNN, Dynamic-kNN and Dynamic-VLM. Static-kNN employs the k-nearest neighbor algorithm to select the top-k nearest objects at the beginning of each episode, with this selection remaining fixed for the duration of the rollout. In contrast, Dynamic-kNN recalculates the k-nearest objects at specified intervals during the policy execution. Notably, both of these techniques rely solely on the metric properties of the objects to identify relevant ones. Dynamic-VLM, on the other hand, uses a vision-language model to choose relevant objects based on both metric and semantic features. At a certain time step t, Dynamic-VLM takes as input a sequence of past images $\mathcal{I}_{t-\Delta t,t}$ (Fig. 3), the names of all N objects in the scene $\mathcal{O} = \{o_i\}_{i=1}^{N}$, the task prompt P and the number of relevant objects to select N_{rel} and outputs the set of relevant objects $\mathcal{O}_{\text{rel}} = \{o_i\}_{i=1}^{N_{\text{rel}}}$. Each image is annotated with object bounding boxes and names as shown in Fig. 3. Additionally, the path followed by the end-effector $[\mathbf{x}_0^{\text{EE}}, \ldots, \mathbf{x}_t^{\text{EE}}]$ up to the current time t is projected and overlaid on the image. We use GPT-4o as our vision-language model and leverage its structured output feature to constrain the output to the provided set of object names $\mathcal{O} = \{o_i\}_{i=1}^{N}$. Dynamic-VLM selects objects based on their metric properties such as proximity (objects which are more likely to come on contact), as well as their semantic properties (fragile objects are safety critical, whereas durable objects can be ignored).

Constraint Selection: Given a task description and the current scene, we use a vision-language model to identify relevant constraints $\mathcal{T}_{\text{rel}}, \mathcal{F}_{\text{rel}}$ from the full sets of valid constraints $\mathcal{T}, \mathcal{F}$. We exploit the semantic reasoning capabilities of VLMs to identify constraints for objects and object pairs. To enable that, we first convert the set of geometric constraints $\mathcal{T}, \mathcal{F}$ to semantic constraints $\mathcal{T}^{\text{sem}}, \mathcal{F}^{\text{sem}}$ in natural language. For instance, the relative velocity constraint that allows soft contacts between two interacting objects $g(\mathbf{s}_{\text{book}}, \mathbf{s}_{\text{toy}}) = \epsilon - ||\dot{\mathbf{x}}_{\text{book}} - \dot{\mathbf{x}}_{\text{toy}}||$ can be semantically written as $g^{\text{sem}}(\texttt{<book>}, \texttt{<toy>}) = \texttt{<soft contact>}$. We name our VLM-based constraint selection method, *ContraintVLM*, and task it to identify the semantic safety constraints for all object pairs such that $\text{ContraintVLM}(o_i, o_j, \mathcal{F}^{\text{sem}}) = g^{\text{sem}}(o_i, o_j)$ where $o_i, o_j \in \mathcal{O}_{\text{rel}}$ and $g^{\text{sem}} \in \mathcal{F}^{\text{sem}}$. Once the set of semantic constraints are selected by the VLM, they can be converted back to their geometric formulations and utilized by the parameterized policy. The contraints are identified at the beginning of each episode ($t = 0$) and kept fixed for the remainder of the rollout.

V. Experimental Setup

We design our experiments to answer the following research questions:

- **Q1:** Does the discounted reach-avoid formulation of the RL objective help learn safe policies that consider safe dynamic interactions?
- **Q2:** In cluttered tabletop manipulation environments, can dimensionality reduction techniques that identify relevant

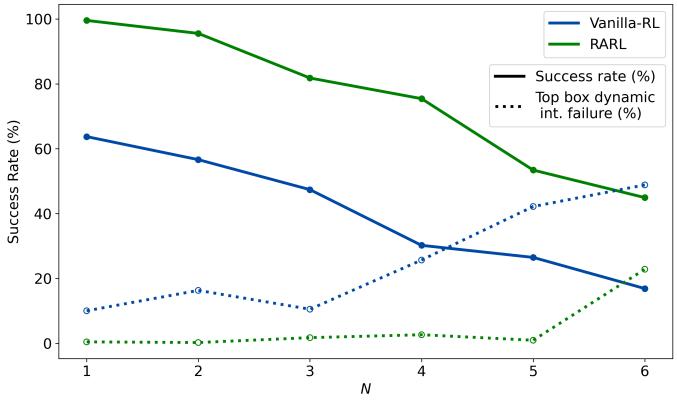

Figure 4: Comparison of success rates and top box safety violations between Vanilla-RL and RARL across varying numbers of cluttered objects on the table.

safety-critical objects yield safe policies while preserving performance?

- **Q3:** Can VLMs effectively infer semantic safety criteria from image observations and semantic descriptions of the task?

Task description. We consider a tabletop manipulation setting where the robot must interact with everyday objects such as cereal boxes, porcelain mugs, stuffed toys, and more (Fig. 1). To test the *dynamics-aware* safety properties of our approach, we specifically instantiate a challenging dynamic task, where the robot is required to dynamically, but carefully, pull a cereal box from under another cereal box on a cluttered tabletop, and safety place it in a goal region. We simulate robot and environment dynamics with MuJoCo [28] and use the Google Scanned Objects Dataset [12, 13, 33] to simulate everyday objects in the table arena defined in Robosuite [37]. The full state for this task is as follows: $\mathbf{s} = [\mathbf{s}^{\text{EE}}, \mathbf{s}^{\text{cbot}}, \mathbf{s}^{\text{ctop}}, \mathbf{s}^o_1, \ldots, \mathbf{s}^o_N]$, where $\mathbf{s}^{\text{cbot}}$ is the state of the bottom cereal box, $\mathbf{s}^{\text{ctop}}$ is the state of the top cereal box and $\mathbf{s}^o_i$ is the state of the i^{th} object in clutter on the table. N is the total number of cluttered objects on the table, that is, in addition to the two cereal boxes. We denote N_{rel} as the number of relevant cluttered objects, which is in *addition* to the two cereal boxes, that are always considered relevant since they are essential to the task.

Target and failure set. The target set $\mathcal{T}$ for this task is defined by the following task objectives: 1) the bottom cereal box should be completely slid out from under the top cereal box $l_1(\mathbf{s}_t) = ||\mathbf{x}^{\text{cbot}}_t - \mathbf{x}^{\text{ctop}}_t|| - d^1_{\text{thresh}}$, 2) the bottom cereal block is slid to a goal region near the end of the table $l_2(\mathbf{s}_t) = d^2_{\text{thresh}} - ||\mathbf{x}^{\text{cbot}}_t - \mathbf{x}^{\text{goal}}||$. For the task to be completed, both constraints must be satisfied such that $l(\mathbf{s}_t) = \min(l_1(\mathbf{s}_t), l_2(\mathbf{s}_t)) > 0$. The failure set $\mathcal{F}$ is comprised of the following safety constraints: 1) the top cereal block should not be aggressively displaced from its initial position $g_{\text{dyn}}(\mathbf{s}_t) = d^3_{\text{thresh}} - ||\mathbf{x}^{\text{ctop}}_{t=t} - \mathbf{x}^{\text{ctop}}_{t=0}||$, for instance, the top cereal box should not be flipped over. This constraint captures the dynamic interactions between the bottom and top cereal boxes, which is a key consideration for dynamic manipulation tasks, 2) the bottom cereal box *should not* make contact with any fragile object $g_{\text{hard}}(\mathbf{s}_t) = ||\mathbf{x}^{\text{cbot}}_t - \mathbf{x}^{\text{o}}_t|| - d^4_{\text{thresh}}$. This is a hard collision avoidance constraint. However, since this is a dynamic task (not quasi-static), long-horizon reasoning is required to avoid contacts, 3) the bottom cereal box should only make soft/gentle contacts with non-fragile/soft objects in clutter $g_{\text{soft}}(\mathbf{s}_t) = d^{\text{vel}}_{\text{thresh}} - ||\dot{\mathbf{x}}^{\text{cbot}}_t - \dot{\mathbf{x}}^{\text{o}}_t||$. This is a relative velocity constraint, which again requires long-horizon reasoning, so that the cereal box can slow down before making contact with the object.

Baselines. To answer **Q1**, we compare the performance of the reach-avoid safety policy against vanilla DDPG with Mean-squared Bellman update. We call this baseline **Vanilla-RL**. To answer **Q2** and **Q3**, we train a policy with full-state privileged information ($N_{\text{rel}} = N$) and compare its performance with our reduced state policies using different relevant object selection techniques (Sec. IV-C). Furthermore, we analyze the effect of the choice of N_{rel} for varying number of total objects N. To answer **Q3**, we compare the performance of ConstraintVLM, which leverages semantic information to identify safety constraints, against hand-designed conservative heuristics.

Training Details. For policy learning, both RARL and Vanilla-RL use the same target and failure sets, however, Vanilla-RL uses the traditional expected sum of rewards formulation while RARL uses the min-over-time formulation as described in Sec. IV-B. Both methods are optimized via DDPG [20] and have the same model architecture and training hyperparameters (for details refer to Appendix A). Both actor and critic networks are three-layer MLPs, each with 256 units and ReLU activation. Wall time for training is around 15 hours. The policies are trained using one NVIDIA RTX A6000 GPU.

VLM for Dimensionality Reduction and Constraint synthesis. We use GPT-4o [24] as the vision-language model for our experiments. During execution, Dynamic-kNN and Dynamic-VLM are queried every four timesteps to update the relevant object set. Images taken at previous two queries are appended together as shown in Fig. 3, overlaid with bounding boxes and past trajectory, and sent as input to Dynamic-VLM. ContraintVLM is queried once at the beginning of each episode to get the constraint types. Full prompt for object selection is provided in Appendix B and for constraint synthesis is provided in Appendix C.

VI. Simulation Results

A. RARL generates safer control policies that consider safe dynamic interactions

Methods. We train separate policies for different number of cluttered objects on the table ($N = 1, 2, 3, 4, 5, 6$) using both Vanilla-RL and RARL. We evaluate these policies in the same setting as during training, i.e. with the same number of total objects. All above policies are trained considering the target set $\mathcal{T}$ explained in Sec. V and failure set consisting of 1) dynamic

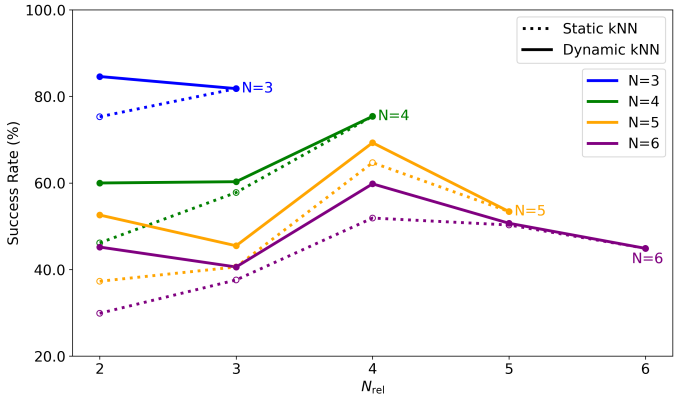

Figure 5: Comparing success rates of Static-kNN vs. Dynamic-kNN for different values of N_{rel} and N.

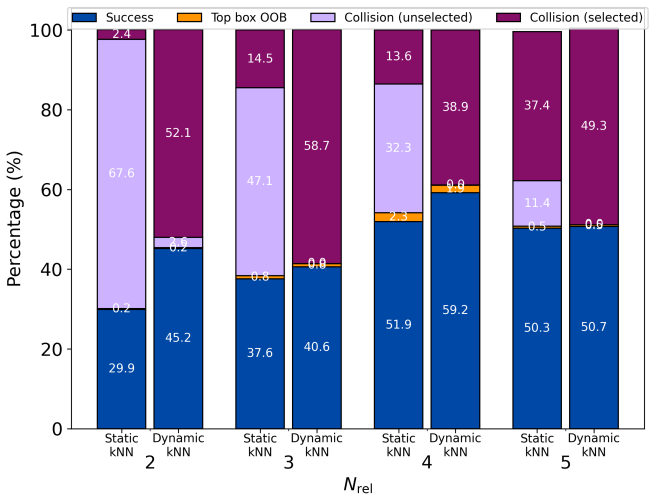

Figure 6: Percentage success and failure for Static-kNN vs. Dynamic-kNN. The three failure scenarios are 1) top block out of safety bounds (OOB), 2) collision with unselected objects, 3) collision with selected (relevant) objects. Results are presented for $N = 6$.

safety constraint for top cereal box (g_{dyn}), and 2) no collision constraint between the bottom cereal box and all objects in clutter (g_{hard}).

Metrics. To compare the performance of RARL vs. Vanilla-RL policies, we consider the following metrics: 1) Success rate (%), and 2) Safety violation rate (%). We report average numbers across 2000 rollouts.

Results: Vanilla-RL vs. RARL. Figure 4 compares the success rates of Vanilla-RL and RARL for varying number of cluttered objects (N) on the table (solid lines). We note that RARL consistently outperforms Vanilla-RL. Furthermore, as the number of objects (N) increases the performance of both methods deteriorates. This is because, as the dimensionality of the state space increases, without increase in the model size, policy learning using RL becomes increasingly challenging. The two failure modes in these experiments are: 1) top cereal box leaving the safety set $g_{\text{dyn}}(\mathbf{s}_t) < 0$ due to dynamic interactions with the bottom cereal box, and 2) collision of the manipulated (bottom) cereal box with any of the obstacles on the table $g_{\text{hard}}(\mathbf{s}_t) < 0$. Figure 4 plots the rate of safety violation of the top cereal box $g_{\text{dyn}}(\mathbf{s}_t) < 0$ (dotted lines), which is a percentage of the total rollouts. We observe that RARL consistently has much lower constraint violations, showing the strength of this method for effectively handing dynamic interactions.

B. Dimensionality reduction techniques yield safe policies while preserving performance by identifying relevant safety-critical objects.

Methods. We compare various dimensionality reduction techniques (Sec. IV-C) against policies trained with full-state privileged information ($N_{\text{rel}} = N$). Additionally, we analyze policy performance across varying choices of N_{rel} in environments of different complexity N. To achieve this, we train a set of policies with different capacities $N_{\text{rel}} = 2, 3, 4, 5, 6$ and test them in environments with different complexities $N = 3, 4, 5, 6$. All above policies are trained with the target set $\mathcal{T}$ defined in Sec. V and a failure set comprising 1) dynamic safety constraints for the top cereal box (g_{dyn}), and 2) no-collision constraints between the bottom cereal box and all clutter objects (g_{hard}). We call this a conservative failure set $\mathcal{F}^{\text{conser}}$ and the trained policy is a conservative policy $\pi^{\text{conser}}_{N_{\text{rel}}}$, where all N_{rel} objects are considered *fragile* (no-contact constraint).

Metrics. We evaluate the performance of different relevant object detection techniques using the following metrics: 1) Success rate (%), 2) Safety violation rate (%), 3) Rate of collision with unselected objects i.e. objects that were considered irrelevant. The last metric is used to infer the proficiency of each of the object selection techniques at choosing the *correct* relevant object.

Results: Impact of dimensionality reduction for varying N_{rel} in environments of varying complexity N, assuming uniform semantics. We consider the scenario where all objects are assumed to have uniform semantics (fragile) and collision needs to be avoided with all objects $\mathcal{F}^{\text{conser}}$. Under this assumption, metric features (position/velocity) are sufficient for identifying relevant objects. Thus, we compare the two object selection methods, Static-kNN and Dynamic-kNN, that rely purely on metric properties of objects. Fig. 5 presents success rates of pretrained policies $\pi^{\text{conser}}_{N_{\text{rel}}}$, for different choices of N_{rel}, tested in environments with varying complexity (N). In settings where $N_{\text{rel}} = N$, the policy $\pi^{\text{conser}}_{N_{\text{rel}}=N}$ is a privileged policy with full-state information.

From Fig. 5, we observe that Dynamic-kNN consistently outperforms Static-kNN, as expected. This is because, as the task progresses and the cereal box is slid along the table, the set of relevant objects (those to avoid collision with) changes, something that Static-kNN fails to account for. We further note that, as N increases, the performance of the privileged policy $\pi^{\text{conser}}_{N_{\text{rel}}=N}$ suffers, and choosing a policy

trained with smaller state space $\pi^{\text{conser}}_{N_{\text{rel}}<N}$ and considering a subset of safety-critical objects becomes more beneficial. This can be especially noted for scenarios with $N = 5, 6$ where $\pi^{\text{conser}}_{N_{\text{rel}}=2}$ performs comparable to, and $\pi^{\text{conser}}_{N_{\text{rel}}=4}$ outperforms, the privileged policy. It is interesting to note that, for a fixed N we do not necessarily observe a consistent increase in performance as we choose policies with increasing capacity N_{rel}. This is because there is a trade off between complexity and representation capacity of $\pi^{\text{conser}}_{N_{\text{rel}}}$. Policies trained with higher N_{rel} have higher representation capacity and avoid contacts with more objects, however, since these policies are trained in more complex environments, with the same model size, they are less performant.

We further breakdown the failure cases for Static-kNN and Dynamic-kNN in Fig. 6. In this scenario, there are three failure modes: 1) top cereal box leaving safety set or out of bound (OOB), 2) collision with unselected objects (objects deemed irrelevant), and 3) collision with selected objects (objects deemed relevant). We observe that Dynamic-kNN has significantly fewer collisions with objects that it deemed irrelevant, as compared to Static-kNN, demonstrating its capability to select safety-critical objects with higher accuracy.

Results: VLM vs. proximity-based dimensionality reduction methods with metric-semantic safety constraints. We populate the environment with a collection of fragile as well as durable objects. Thus, the failure set for this setting is defined such that collision should be avoided with the *fragile* objects while other *durable* objects can be freely interacted with. Hence, only a subset of objects are safety critical and others can be ignored. We compare the performance of Static-kNN, Dynamic-kNN and Dynamic-VLM for selecting relevant safety critical objects. We also compare against a privileged selection heuristic that always chooses the k-nearest *fragile* objects. Policy π^{conser} is executed and, as a reminder, since it is trained considering $\mathcal{F}^{\text{conser}}$, it will try to avoid all selected objects regardless of their semantic properties. In Table I, we report success rates (%) in an environment with five objects in clutter (N=5) and we choose two relevant objects $N_{\text{rel}} = 2$. We note that Dynamic-VLM outperforms both Static-kNN and Dynamic-kNN. This is because Static-kNN and Dynamic-kNN only consider the metric features of objects and will always choose the k-nearest objects regardless of their semantic features. Dynamic-VLM, on the other hand, considers both metric and semantic features and prioritizes selecting fragile objects since they are safety critical. The privileged selection mechanism performs the best as it always selects the nearest fragile objects. Naturally, performance across all techniques worsens as the number of fragile objects in the scene increases.

C. VLMs can infer semantic safety criteria from semantic observations and task description.

Methods. We compare the performance of different dimensionality reduction techniques in a more complex setting, where each object can have different semantic properties, *fragile*, *soft*, or *durable*, and hence a different constraint type. The failure set $\mathcal{F}^{all}$ consists of states that satisfy, 1) dynamic interaction constraint g_{dyn}, 2) *no contact* constraint with fragile objects g_{hard}, 3) *soft contact* constraint with soft objects g_{soft} and free collisions with durable objects. We train policy $\pi^{\text{all}}_{N_{\text{rel}}}$ for all combinations of constraints mentioned above for each cluttered object in the scene. We parameterize the policy with the semantic properties of objects from which the constraint type can be inferred. Specifically, the object state is appended with an integer corresponding to its semantic property (fragile $= 0$, soft $= 0$, or durable $= 2$). During execution, the constraint types are inferred using *ConstraintVLM* which selects from a set of semantic constraint types {`<no contact>`, `<soft contact>`, `<any contact>`} for each object. We compare against approaches where Static-kNN and Dynamic-kNN are used for relevant object selection, however, the constraint types are manually designed and assumed to be conservative, i.e. `<no contact>` for all objects.

Table I: Comparison of dimensionality reduction techniques in environments with semantically different objects (fragile and durable) using a pretrained conservative policy $\pi^{\text{conser}}_{N_{\text{rel}}=2}$. Total number of objects $N = 5$. Number of selected objects $N_{\text{rel}} = 2$. Table reports success rate (%).

Object semantic types	Static-kNN	Dynamic-kNN	Dynamic-VLM	Privileged selection
2 fragile, 3 durable	61.3	66.8	80.8	91.6
3 fragile, 2 durable	46.9	54.8	63.0	69.1

Table II: Comparison of in scenes with semantically different objects (fragile, soft, durable). Table reports success rate (%).

Object selection Constraint selection	Static-kNN Conservative	Dynamic-kNN Conservative	Dynamic-VLM ContraintVLM	Dynamic-kNN ContraintVLM
2 fragile, 2 soft, 1 durable ($N_{\text{rel}} = 2$)	45.3	53.6	57.2	51.8
2 fragile, 2 soft, 2 durable ($N_{\text{rel}} = 3$)	44.7	49.2	53.3	50.0

Results. Table II shows the performance when using Constraint-VLM for constraint selection for different choices of object selection methods. We observe that using Constraint-VLM for constraint selection and Dynamic-VLM for object selection achieves the highest performance since VLMs are able to better capture the semantic properties of relevant objects. Moreover, when assuming 'conservative' constraints for all objects, the policy follows suboptimal paths to avoid all nearby objects, leading to low task completion rates. To quantitatively analyze the proficiency of VLMs at identifying appropriate constraint types, based on the semantic properties of objects, we construct a confusion matrix. Table III shows

Table III: Confusion matrix for constraint type prediction using ConstraintVLM. Percentage of total trials (%).

Constraint types	`no contact`	`soft contact`	`any contact`
`no contact` (GT)	99.0	1.0	0.0
`soft contact` (GT)	25.2	41.5	0.0
`any contact` (GT)	0	0	100

Table IV: Constraint type prediction using ConstraintVLM for different objects. Percentage of total trials (%).

Object	no contact	soft contact	any contact
toy squirrel	0.0	0.0	100.0
toy sheep	0.0	0.0	100.0
toy android	0.0	0.0	100.0
red mug	98.5	1.5	0.0
blue mug	98.5	1.5	0.0
porcelain mug	100.0	0.0	0.0
supplement	6.6	93.4	0.0
plant pot	73.0	27.0	0.0

the percentage by which each ground truth constraint is correctly identified. We note that `<no contact` (fragile) and `<any contact` (durable) constraints are identified with high reliability, while soft objects can be confused to be fragile. Furthermore, table IV shows constraint type predictions for different objects. Both tables are constructed using 200 trails. We note that for fragile objects like mugs, ContraintVLM predicts `no contact` constraints, while for durable objects like toys it predicts `any contact` constraints, while semi-fragile objects are allowed soft contacts. This aligns well with our ground truth treatment of these objects.

VII. Limitations

Our approach has several limitations. First, the number of relevant objects for a task must be predetermined and is kept fixed during deployment, reducing adaptability. Future work will explore dynamically changing N_{rel} as the task proceeds. Training parameterized policies for all possible constraint combinations in an RL setting is computationally expensive, requiring high-fidelity simulation for control policy synthesis and sim-to-real transfer for real-world deployment. Another challenge is the slow inference of Vision-Language Models (VLMs), making them unsuitable for high-frequency real-world applications due to latency, certification, and reliability concerns. To address this, we plan to use language models at lower frequencies to update relevant objects and constraints. Finally, our method assumes access to ground-truth object states, and future work will investigate leveraging 3D semantic representations to mitigate this reliance.

VIII. Conclusion

We propose a scalable method for learning safe policies for dynamic and interactive manipulation in cluttered environments using reach-avoid reinforcement learning. Our approach includes an offline training pipeline that focuses on a reduced set of objects while incorporating diverse task and safety constraints. During execution, we employ heuristic-based and vision-language based techniques to dynamically identify relevant objects and constraints based on both metric and semantic properties. Simulation results on a complex manipulation task demonstrate that task- and safety-aware dimensionality reduction enables efficient policy learning while preserving safety guarantees, supporting deployment in unstructured environments.

References

[1] Fares J Abu-Dakka and Matteo Saveriano. Variable impedance control and learning—a review. *Frontiers in Robotics and AI*, 7:590681, 2020.

[2] Fares J Abu-Dakka, Leonel Rozo, and Darwin G Caldwell. Force-based learning of variable impedance skills for robotic manipulation. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–9. IEEE, 2018.

[3] Anayo K Akametalu, Shromona Ghosh, Jaime F Fisac, Vicenc Rubies-Royo, and Claire J Tomlin. A minimum discounted reward hamilton-jacobi formulation for computing reachable sets. *IEEE Transactions on Automatic Control*, 2023.

[4] Brian DO Anderson and John B Moore. *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[5] Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2242–2253. IEEE, 2017.

[6] Richard Bellman, Robert E Kalaba, et al. *Dynamic programming and modern control theory*, volume 81. Citeseer, 1965.

[7] Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 2012.

[8] Lukas Brunke, Yanni Zhang, Ralf Römer, Jack Naimer, Nikola Staykov, Siqi Zhou, and Angela P Schoellig. Semantically safe robot manipulation: From semantic scene understanding to motion safeguards. *arXiv preprint arXiv:2410.15185*, 2024.

[9] Aidan Curtis, Nishanth Kumar, Jing Cao, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Trust the proc3s: Solving long-horizon robotics problems with llms and constraint satisfaction. *arXiv preprint arXiv:2406.05572*, 2024.

[10] Jérôme Darbon and Stanley Osher. Algorithms for overcoming the curse of dimensionality for certain hamilton–jacobi equations arising in control theory and elsewhere. *Research in the Mathematical Sciences*, 3(1):19, 2016.

[11] Alessandro De Luca and Lorenzo Ferrajoli. Exploiting robot redundancy in collision detection and reaction. In *2008 IEEE/RSJ international conference on intelligent robots and systems*, pages 3299–3305. IEEE, 2008.

[12] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022.

[13] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google scanned

objects: A high-quality dataset of 3d scanned household items, 2022. URL https://arxiv.org/abs/2204.11918.

[14] Jaime F Fisac, Neil F Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J Tomlin. Bridging hamilton-jacobi safety analysis and reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8550–8556. IEEE, 2019.

[15] Neville Hogan. Impedance control: An approach to manipulation. In *1984 American control conference*, pages 304–313. IEEE, 1984.

[16] Kai-Chieh Hsu, Vicenç Rubies-Royo, Claire J Tomlin, and Jaime F Fisac. Safety and liveness guarantees through reach-avoid reinforcement learning. *arXiv preprint arXiv:2112.12288*, 2021.

[17] Nishanth Kumar, Fabio Ramos, Dieter Fox, and Caelan Reed Garrett. Open-world task and motion planning via vision-language model inferred constraints. *arXiv preprint arXiv:2411.08253*, 2024.

[18] Zhaoting Li, Miguel Zamora, Hehui Zheng, and Stelian Coros. Embracing safe contacts with contact-aware planning and control. *arXiv preprint arXiv:2308.04323*, 2023.

[19] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[20] TP Lillicrap. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2016.

[21] Kostas Margellos and John Lygeros. Hamilton–jacobi formulation for reach–avoid differential games. *IEEE Transactions on automatic control*, 56(8):1849–1861, 2011.

[22] Roberto Martín-Martín, Michelle A Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1010–1017. IEEE, 2019.

[23] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on automatic control*, 50(7):947–957, 2005.

[24] et al OpenAI Team. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

[25] Luka Peternel, Tadej Petrič, and Jan Babič. Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 1497–1502. IEEE, 2015.

[26] Hamid Sadeghian, Luigi Villani, Mehdi Keshmiri, and Bruno Siciliano. Task-space control of robot manipulators with null-space compliance. *IEEE Transactions on Robotics*, 30(2):493–506, 2013.

[27] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

[28] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

[29] Jake Varley, Sumeet Singh, Deepali Jain, Krzysztof Choromanski, Andy Zeng, Somnath Basu Roy Chowdhury, Avinava Dubey, and Vikas Sindhwani. Embodied ai with two arms: Zero-shot learning, safety and modularity. *arXiv preprint arXiv:2404.03570*, 2024.

[30] Huaxiaoyue Wang, Nathaniel Chin, Gonzalo Gonzalez-Pumariega, Xiangwan Sun, Neha Sunkara, Maximus Adrian Pace, Jeannette Bohg, and Sanjiban Choudhury. Apricot: Active preference learning and constraint-aware task planning with llms. *arXiv preprint arXiv:2410.19656*, 2024.

[31] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

[32] Ziyi Yang, Shreyas S Raman, Ankit Shah, and Stefanie Tellex. Plug in the safety chip: Enforcing constraints for llm-driven robot agents. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14435–14442. IEEE, 2024.

[33] Kevin Zakka. Scanned Objects MuJoCo Models, 7 2022. URL https://github.com/kevinzakka/mujoco_scanned_objects.

[34] Xiang Zhang, Changhao Wang, Lingfeng Sun, Zheng Wu, Xinghao Zhu, and Masayoshi Tomizuka. Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning. In *Conference on Robot Learning*, pages 1621–1639. PMLR, 2023.

[35] Xiang Zhu, Shucheng Kang, and Jianyu Chen. A contact-safe reinforcement learning framework for contact-rich robot manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2476–2482. IEEE, 2022.

[36] Xinghao Zhu, Wenzhao Lian, Bodi Yuan, C Daniel Freeman, and Masayoshi Tomizuka. Allowing safe contact in robotic goal-reaching: Planning and tracking in operational and null spaces. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8120–8126. IEEE, 2023.

[37] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, Yifeng Zhu, and Kevin Lin. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

APPENDIX

A. Training details

Table V: Training hyperparameters

Hyperparameter	Value
Batch size	200
# Training steps	3M
Learning rate	0.001
Max. episode length	300
Optimizer	AdamW
Replay buffer size	100k
Discount factor	0.99
Polyak	0.995
Relative velocity threshold	1.0 m/s
Collision threshold	0.01 m

B. Prompt for relevant object selection

Object selection prompt: You are an excellent safe planning agent for dynamic tasks. You are given a task description and a sequence of images showing the trajectory followed by the robot and the objects so far, the last image shows the current state. The objects are highlighted using bounding boxes with labels `<object id>: <object name>`. The past trajectory of the robot can be inferred from the sequence of images and also from the black arrow showing the past path followed by the robot end effector. Important note: the black line with the arrow shows the past trajectory already followed by the end-effector in the past few time steps, not the planned future trajectory. Expecting that the robot will continue along the path, try to predict the future trajectory and choose relevant objects in the future: You need to do this based on two criteria: proximity and safety. Under the 'proximity' criteria you consider the objects which are most likely to come in contact with the top cereal box or end-effector if the robot continues along its its trajectory in the future. The object list provided to you is sorted in increasing order of the distance between the end-effector and the object. For example, object with index 0 will be closest to the end-effector, object with index 1 will be further away and so on. Use this list to help with the proximity criteria. Under the 'safety' criteria, among the objects that might come in contact in the near future, choose objects which are more fragile/safety critical, and coming in contact with them is more risky. Provide explanation for why you are choosing these objects (proximity vs safety) and why they are relevant in the near future. In 'image_description' briefly describe 1) brief description of the sequence of images, 2) describe the trajectory followed by the robot (black arrow) so far 3) describe what the future trajectory will look like 4) which objects might likely come in contact with the gripper in the future.
Task prompt: A robot arm needs to slide the white cereal box from under the blue cereal box and to the right, without damaging other objects along the way.

C. Prompt for constraint selection

Constraint selection prompt: You are an excellent safe planning agent for dynamic tasks. You are given a task description and a list of objects on a table. You need to decide the safe interaction type for each object from the list of constraint types. Here the description of the constraint types: 'no_contact' implies that there should absolutely be no contact with a certain object. 'soft_contact' implies that you can softly interact with that object, push it softly, etc. 'any_contact' implies that any kind of interaction including aggressive impact is allowed. Some hints on how to decide on the constraint type for an object: If an object is soft or made of durable material, and softly pushing it or moving it without toppling it is okay, 'soft_contact' can be allowed with that object. If an object is very durable, and pushing it aggressively will not damage it, 'any_contact' can be allowed with that object. If an object is fragile, and contacting it might damage it, 'no_contact' should be allowed with that object. Usually objects such as cups, wine glasses, bowls, electronics, etc are considered fragile and should be 'no_contact'. Plastic objects such as bottles, plastic cans, tubes can be allowed 'soft_contact'. Soft and non-critical objects such as toys, clothing, etc are soft and can be ignored and allowed 'any_contact'. Do not respond on the basis of whether given the task, contact will be needed or not, but respond based on what kind of interaction is safe with a particular object regardless of the task. Provide brief explanation, for choosing a specific constraint type for an object.
Task prompt: A robot arm needs to slide the white cereal box from under the blue cereal box and to the right, without damaging other objects along the way.