2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Robotic Table Wiping via Reinforcement Learning and Whole-body Trajectory Optimization

Thomas Lew[1,3], Sumeet Singh[1], Mario Prats[2], Jeffrey Bingham[2], Jonathan Weisz[2], Benjie Holson[2], Xiaohan Zhang[1,4], Vikas Sindhwani[1], Yao Lu[1], Fei Xia[1], Peng Xu[1], Tingnan Zhang[1], Jie Tan[1], Montserrat Gonzalez[1]

***Abstract*— We propose a framework to enable multipurpose assistive mobile robots to autonomously wipe tables to clean spills and crumbs. This problem is challenging, as it requires planning wiping actions while reasoning over uncertain latent dynamics of crumbs and spills captured via high-dimensional visual observations. Simultaneously, we must guarantee constraints satisfaction to enable safe deployment in unstructured cluttered environments. To tackle this problem, we first propose a stochastic differential equation to model crumbs and spill dynamics and absorption with a robot wiper. Using this model, we train a vision-based policy for planning wiping actions in simulation using reinforcement learning (RL). To enable zero-shot sim-to-real deployment, we dovetail the RL policy with a whole-body trajectory optimization framework to compute base and arm joint trajectories that execute the desired wiping motions while guaranteeing constraints satisfaction. We extensively validate our approach in simulation and on hardware.**

Video of experiments: `https://youtu.be/inORKP4F3EI`

I. Introduction

Multipurpose assistive robots will play an important role in improving people's lives in the spaces where we live and work [1], [2]. Repetitive tasks such as cleaning surfaces are well-suited for robots, but remain challenging for systems that typically operate in structured environments. Operating in the real world requires handling high-dimensional sensory inputs and dealing with the stochasticity of the environment.

Learning-based techniques such as reinforcement learning (RL) offer the promise of solving these complex visuo-motor tasks from high-dimensional observations. However, applying end-to-end learning methods to mobile manipulation tasks remains challenging due to the increased dimensionality and the need for precise low-level control. Additionally, on-robot deployment either requires collecting large amounts of data [3]–[5], using accurate but computationally expensive models [5], or on-hardware fine-tuning [6].

In this work, we focus on the task of cleaning tables with a mobile robotic manipulator equipped with a wiping tool. This problem is challenging for both high-level planning and low-level control. Indeed, at a high-level, deciding how to best wipe a spill perceived by a camera requires solving a challenging planning problem with stochastic dynamics. At a low-level, executing a wiping motion requires simultaneously maintaining contact with the table while avoiding nearby obstacles such as chairs. Designing a real-time and effective solution to this problem remains an open problem [1].

[1]Robotics at Google [2]Everyday Robots
[3]Department of Aeronautics and Astronautics, Stanford University
[4]Department of Computer Science, SUNY Binghamton

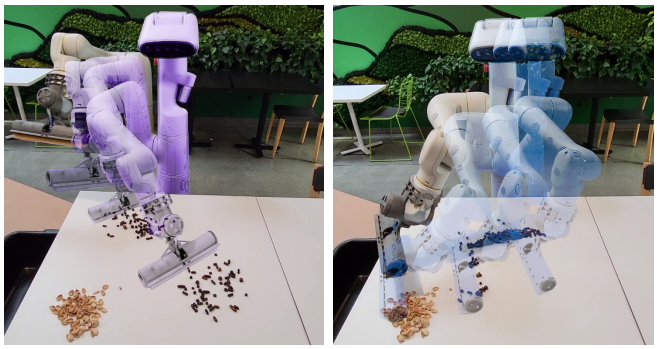

Fig. 1: We present a framework to autonomously clean tables with a mobile manipulator. Our proposed approach combines reinforcement learning to select the best wiping strategy and trajectory optimization to safely execute the wiping actions. We validate our approach on the multipurpose assistive robot from Everyday Robots.

Our main contributions are as follows:

- We propose a framework for autonomous table wiping. First, we use visual observations of the table state to plan high-level wiping actions for the end-effector. Then, we compute whole-body trajectories that we execute using admittance control. This approach is key to achieving reliable table wiping in new environments without the need for real-world data collection or demonstrations.
- We describe the uncertain time evolution of dirty particles on the table using a stochastic differential equation (SDE) capable of modeling absorption with the wiper. Then, we formulate the problem of planning wiping actions as a stochastic optimal control problem. As this task requires planning over high-dimensional visual inputs, we solve the problem using RL, *entirely in simulation*.
- We design a whole-body trajectory optimization algorithm for navigation in cluttered environments and table wiping. Our approach accounts for the kinematics of the manipulator, the nonholonomic constraints of the base, and collision avoidance constraints with the environment.

This approach combines the strengths of reinforcement learning - planning in high-dimensional observation spaces with complex stochastic dynamics, and of trajectory optimization - guaranteeing constraints satisfaction while executing whole-body trajectories; it does not require collecting a task-specific dataset on the system, and transfers zero-shot to hardware.

II. Related work

Reinforcement learning allows tackling complex high-dimensional planning problems with stochastic multimodal dynamics that would be difficult to solve in real-time with

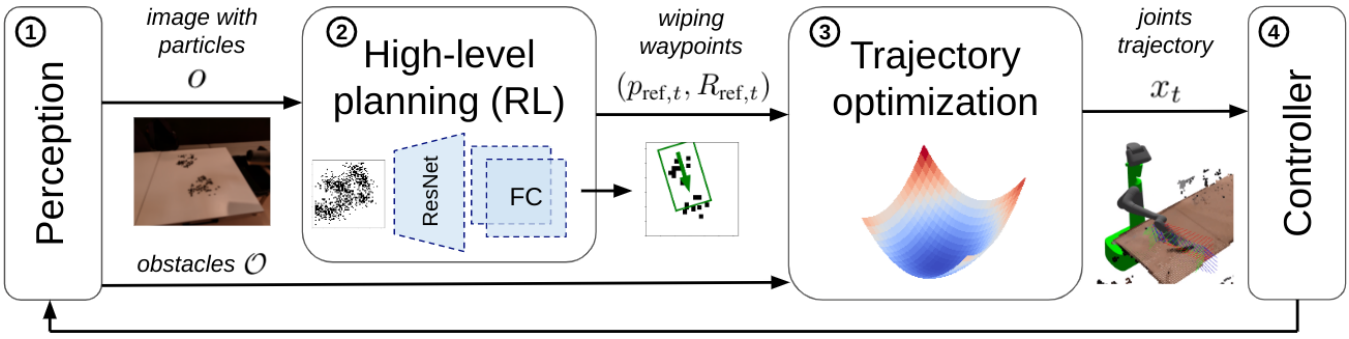

Fig. 2: System overview. (1) A perception module processes sensory inputs (camera images and LiDAR pointclouds) and senses obstacles, crumbs, and spills. (2) A high-level planning module selects wiping waypoints on the table. (3) A whole-body trajectory optimization module computes joint angles to perform the wipe while satisfying constraints. (4) An admittance controller executes the planned trajectory.

model-based techniques [7]–[10]. The success of RL in complex robotic tasks hinges on appropriately selecting the observation and action spaces to simplify learning [11]–[14]. Indeed, end-to-end training is computationally costly and requires expensive data collection [3]–[5]. Previous work demonstrated that decomposing complex problems by planning high-level waypoints with RL and generating motion plans with model-based approaches improves performance [13], [15]. We use a similar decomposition in this work. However, while previous approaches were applied to solve navigation or manipulation tasks independently, we demonstrate that RL can be combined with trajectory optimization and admittance control to simultaneously move the base and arm of a mobile manipulator while avoiding obstacles to solve a complex task such as table wiping.

Trajectory optimization allows computing dynamically-feasible trajectories that guarantee reliable and safe execution, e.g., by accounting for obstacle avoidance constraints. Multiple works have demonstrated real-time trajectory optimization algorithms for mobile manipulators, e.g., for a ball-balancing manipulator [16], [17] and a legged robot equipped with a robotic arm [18]–[20]. In this work, we demonstrate real-time whole-body collision-free trajectory optimization on the mobile manipulator with an arm with seven degrees of freedom shown in Fig. 1. The robot base has a nonholonomic constraint that makes real-time trajectory optimization challenging. Navigating in a cluttered environment such as a kitchen requires real-time collision avoidance. As in [16], [20], [21], we enforce collision avoidance constraints in the formulation. We assume that all perceived obstacles are given as polyhedrons, as is common in the literature [22]–[24]. The optimization problem is solved using the differentiable shooting-sequential-quadratic-programming (ShootingSQP) method presented in [25].

Table wiping with a multipurpose mobile manipulator is challenging, as this task requires simultaneously reasoning about the cleanliness state of the table, planning optimal wiping actions, and acting accordingly [1]. Table wiping approaches can be divided into three categories. First, classical methods detect spills and subsequently apply pre-defined wiping patterns to clean them [26]. These methods work well but are suboptimal as they do not explicitly reason about the time evolution of the spills at planning time. A second class of methods uses analytical or learned transition models for the cleanliness state of the table and subsequently apply classical planning methods to solve these problems [27]–[29]. These methods were applied to manipulation problems too [30]. However, [27], [29] only consider dirt cleaning tasks, and learning transition dynamics accounting for absorption and the sticking behavior of certain liquids (e.g., honey) may be challenging. A third class of imitation learning methods uses demonstration data to learn a cleaning policy [31]–[36]. These methods are successful since wiping primitives are easier to learn than visual transition models.

Our proposed approach does not require training data from the system. Instead, we train an RL policy that selects high-level wiping waypoints entirely in simulation. The key consists of describing crumbs and spill dynamics with a stochastic differential equation (SDE) [37]. In contrast to existing learned [38] models and the material point method [39], our SDE model does not require training data from the system that may be expensive to collect, is efficient to simulate, allows modeling dry objects, sticking and absorption behavior. We then directly deploy the learned wiping policy to hardware. Wipes are executed using trajectory optimization which guarantees reliable and safe execution.

III. Combining RL and Trajectory Optimization

We consider two table wiping tasks: **gathering crumbs** and **cleaning spills**. A robot equipped with a wiper cannot immediately capture crumbs and clean dirt particles. Instead, one may first gather the crumbs together before using a different method to remove them (e.g., with a vacuum cleaner [27] or by pushing them into a bin, see Section VI). Thus, we formulate the objective of the crumbs-gathering task as moving all crumbs particles to the center of the table. For spills-cleaning, the objective is wiping all spill particles.

The problem of table wiping can be formulated as a POMDP from image inputs to control signals to send to the robot actuators. In this work, we decompose the problem and propose a framework (see Fig. 2) consisting of four steps:

1) **Perception**: The system processes LiDAR pointclouds and camera depth and color images and returns bounding boxes for obstacles $\mathcal{O}$ (see Fig. 5) and an image mask o for spills and crumbs on the table (see Section IV).

2) **High-level planning with RL** (Section IV): The reinforcement learning (RL) policy takes the input image with crumbs or spills on the table and returns high-level wiping waypoints, corresponding to desired start and end wiping poses of the end-effector on the table. This policy is entirely trained in simulation using the stochastic model of spill and crumbs dynamics and is directly deployed on the system without the need to gather demonstration data from the system.

3-4) **Trajectory optimization and control** (Section V): To execute the planned wiping actions, we compute whole-body joint trajectories using trajectory optimization. This approach guarantees the satisfaction of constraints such as avoiding self-collisions and nearby obstacles such as chairs. We track the whole-body trajectory using admittance control, which enables fine tracking while wiping with a normal force satisfying hardware requirements.

As discussed in Sections I and II, this task decomposition leverages the strengths of reinforcement learning while enforcing constraints satisfaction with trajectory optimization. We describe these two components in the next two sections.

IV. Planning wipes with reinforcement learning

A. Simulating spill and crumbs dynamics

Central to our approach is a model describing the evolution of the cleanliness state of the table. It has four key features:

- It is able to describe both dry objects pushed by the wiper and liquids absorbed during wiping.
- It can capture multiple disjoint spills.
- It captures the stochasticity of state transitions.
- It can be efficiently simulated.

We describe the cleanliness state of the table with the variable $s = (s^x, s^y, s^z)$, where $(s^x, s^y) \in \mathbb{R}^2$ denotes a location on the table and $s^z \in \{0, 1\}$ denotes whether the corresponding crumbs or spill particles are on the table ($s^z = 0$) or were wiped and are on the wiper ($s^z = 1$). The state at time t is characterized by a measure μ_t over $\mathbb{R}^2 \times \{0, 1\}$. We denote by $W_t^a \subset \mathbb{R}^2$ the surface of the table covered by the wiper of orientation θ_t at time t, as a function of the wiping action a (see Section IV-B). From time t_i, each particle evolves in time according to the SDE

$$\mathrm{d}s_t = \begin{bmatrix} b_1(s_t, a, t) \\ b_2(s_t, a, t) \\ 0 \end{bmatrix} \mathrm{d}t + \begin{bmatrix} \alpha\sigma_1(s_t, a, t) \\ \alpha\sigma_2(s_t, a, t) \\ 0 \end{bmatrix} \mathrm{d}B_t + \begin{bmatrix} 0 \\ 0 \\ h(s_t, a, t) \end{bmatrix} \mathrm{d}P_t^\lambda, \tag{1}$$

where $t \in [t_i, t_i + T_i]$, $s_{t_i} \sim \mu_{t_i}$, and

- B_t is a standard 2-dimensional Brownian motion modeling the stochastic evolution of spill and crumbs particles,
- P_t^λ is a standard Poisson process of intensity $\lambda > 0$, modeling spill particles absorbed by the wiping tool,
- The coefficients b, σ, and h are defined by

$$b(s_t, a, t) = \mathbf{1}\left\{ \begin{bmatrix} s_t^x \\ s_t^y \\ s_t^i \end{bmatrix} \in W_t^a \cap \mathcal{T},\ s_t^z = 0 \right\} v \begin{bmatrix} \cos(\theta_t) \\ \sin(\theta_t) \end{bmatrix},$$

$$\sigma(s_t, a, t) = \mathbf{1}\left\{ \begin{bmatrix} s_t^x \\ s_t^y \\ s_t^i \end{bmatrix} \in W_t^a \cap \mathcal{T},\ s_t^z = 0 \right\} v \begin{bmatrix} \cos(\theta_t) & -\sin(\theta_t) \\ \sin(\theta_t) & \cos(\theta_t) \end{bmatrix},$$

$$h(s_t, a, t) = \mathbf{1}\left\{ \begin{bmatrix} s_t^x \\ s_t^y \\ s_t^i \end{bmatrix} \in W_t^a \cap \mathcal{T},\ s_t^z = 0 \right\},$$

Fig. 3: Spills and crumbs wiping simulator using the SDE in (1) *with dirty particles in black and wiped region in green. (1) Initial state. (2) With* $(\lambda, \alpha) = (0, 0)$*, we simulate crumbs that are pushed by the wiper. (3) With* $(\lambda, \alpha) = (0, 0.05)$*, we simulate a spill that is pushed on a table and sticks below the wiper. (4) With* $(\lambda, \alpha) = (5, 0.05)$*, we simulate a spill that is partially absorbed by the wiper.*

where $\mathcal{T} = [0, \bar{w}] \times [0, \bar{h}]$ is the table area, $\alpha > 0$ is a diffusion coefficient, and $\mathbf{1}(\cdot)$ is the indicator function.

The SDE (1) implies that the total mass of dirty particles is conserved over time. These particles either move on the table ($s^z = 0$) or are cleaned and move onto the wiper ($s^z = 1$). Only particles that are on the table ($s_t^z = 0$) and are in contact with the wiper ($(s_t^x, s_t^y) \in W_t^a \cap \mathcal{T}$) can move.

We present simulation results in Fig. 3 for a single wiping action. To better represent the density of particles, we convolve the visualized particle representation with a Gaussian filter. For $\lambda \approx 0$, crumbs and dirt pushing behavior is simulated with no particles entering the wiper. By increasing λ, particles are cleaned and absorbed by the wiper.

B. Observation and action spaces

Observations: Since only visual observations are available in practice, we convert the cleanliness state s_t to an image $o(s_t)$ with 64×64 pixels by setting each pixel occupied by a particle to a maximum value. This approach makes transferring to the real system straightforward: only a mask defining the spill locations is necessary to deploy our approach to the robot, thereby minimizing the simulation to real gap.

Actions: We plan for a sequence of wipes $a_i = (p_i^x, p_i^y, \theta_i, \ell_i)$, where (p_i^x, p_i^y, θ_i) denotes the starting position and direction of the wipe at time t_i (assume 0 w.l.o.g.) and ℓ_i is the wipe length. The action space is defined as $\mathcal{A} = [0, \bar{w}] \times [0, \bar{h}] \times [0, 2\pi) \times [0, L]$, where $L = \min(\bar{w}, \bar{h})$. At time t, the wiper covers the surface $W_t^a \subset \mathbb{R}^2$ of the table parameterized by $w_t^a = (w^x, w^y, w^\theta)_t^a$. The wiper moves at a constant speed v according to $w_t^a = (1-t)(p_i^x, p_i^y, \theta_i) + (p_i^x + tv\cos(\theta_i), p_i^y + tv\sin(\theta_i), \theta_i)$ where $t \in [0, T_i]$ with $T_i = \frac{\ell_i}{v}$. This four-dimensional action space makes planning tractable, albeit one could also use pre-defined [26] or learned [31], [34] wiping patterns instead. We then convert a_i to wiping poses $(p_t^{\text{ref}}, R_t^{\text{ref}})$ by aligning the wipe with the pose of the table, see Section VI for details.

C. Wiping objectives and constraints

We define objectives for the tasks defined in Section III.

Gathering crumbs: A simple reward for this problem is

$$R(s, a) = -\left\| (s^x, s^y) - \left(\frac{\bar{w}}{2}, \frac{\bar{h}}{2} \right) \right\|_2 \tag{2}$$

to penalize the spread of dirty particles to the table center.

Cleaning spills can be described as absorbing all spills with the wiper, i.e., maximizing particles with $s^z = 1$. An objective for this task could be maximizing $\mathbb{E}[s^z]$. In

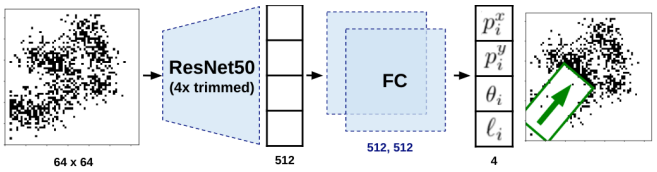

Fig. 4: SAC actor network architecture.

practice, we found that expressing the objective as function of pixel values works better for this task since the latent state s^z is not observable. Thus, we express the objective as

$$R(s,a) = \sum_{jk} (o_{jk}(s_{t+1}) - o_{jk}(s_t)) \tag{3}$$

We do not minimize the wiping lengths ℓ_i, since they do not exactly reflect the wiping duration that depends on the full kinematics of the robot and obstacles in the environment.

Wiping requires keeping crumbs and spill particles on the table. Due to the stochasticity in (1) and the initial particles distribution μ_0, we express this requirement with a joint chance constraint [40] $\mathbb{P}\left((s_t^x, s_t^y) \in \mathcal{T} \text{ for all } t \in [0,T]\right) \geq 1-\delta$ for some probability threshold $\delta \in (0,1)$. Since particles are immobile when not in contact with the wiper, this constraint is equivalent to the joint chance constraint $\mathbb{P}\left((s_{t_i}^x, s_{t_i}^y) \in \mathcal{T} \text{ for all } i = 1,\dots,N\right) \geq 1-\delta$. Using Boole's inequality, a conservative reformulation is given by the expectation constraints $\mathbb{E}\left[\mathbf{1}\{s_{t_i} \notin \text{Table}\}\right] \leq \frac{\delta}{N}$ for all i.

D. Solving the problem via reinforcement learning

By considering a sequence of N wipes and summing the objectives in (2) and (3), we obtain the constrained Markov decision process (CMDP) [41]

$$\sup_{\substack{a_i \in \mathcal{A} \\ i=1,\dots,N}} \mathbb{E}\left[\sum_{i=1}^{N} R(s_{t_i}, a_i)\right] \text{ s.t. (1), } \mathbb{E}\left[\mathbf{1}\{s_{t_i} \notin \mathcal{T}\}\right] \leq \frac{\delta}{N}.$$

where each action a_i only depends on visual observations $\{o(s_{t_j}), j \leq i\}$. This MDP problem structure assumes that visual observations $o(s)$ are expressive enough to reconstruct the state s. Without this assumption, the problem above is a constrained partially-observable MDP (POMDP).

Due to the high-dimensional visual observation space, the complexity of the stochastic dynamics in (1), and the potentially multimodal state distribution, solving the CMDP is challenging. Since simulation is efficient and can be carried out in parallel, we solve the problem using RL. A wide range of RL approaches for CMDPs are available in the literature, ranging from methods using Lyapunov functions to restrict the search to feasible policies [42], [43], using learned dynamics models to predict failures and yield a safe policy [40], [44], and safety filters that modify the actions of a policy to yield constraints satisfaction [45], to name a few. We refer to [46] for a recent survey.

Since the policy can be trained in simulation, for simplicity, we opt for constraints penalization via a Lagrangian relaxation and solve the relaxed problem

$$\sup_{\substack{a_i \in \mathcal{A} \\ i=1,\dots,N}} \mathbb{E}\left[\sum_{i=1}^{N} R(s_{t_i}, a_i) - \mu \mathbf{1}\{s_{t_i} \notin \mathcal{T}\}\right] \text{ s.t. (1),}$$

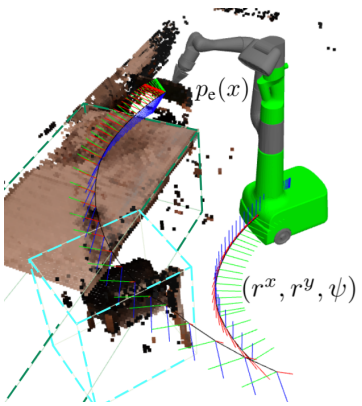

Fig. 5: The robot from Everyday Robots consists of a manipulator with seven joints $(q^1,\dots,q^7)$ mounted on a base with two wheels whose position and yaw is denoted by (r^x, r^y, ψ). The kinematics of the base account for nonholonomic constraints. We enforce collision avoidance constraints within the trajectory optimization formulation using bounding boxes for obstacles.

where $\mu > 0$ is a penalization weight. The problem above can be solved using off-the-shelf RL algorithms. We solve the MDP with the soft actor-critic (SAC) method [47] due to its robustness and sample efficiency and provide further implementation details in Section VI.

V. Executing Wipes with Trajectory Optimization

Next, we describe the whole-body trajectory optimization formulation we use to execute wiping actions with the mobile manipulator shown in Fig. 5. Since the system is passively stable, we plan for kinematically-feasible trajectories. The state of the robot is denoted by $x = (r^x, r^y, \psi, q^1, \dots, q^7)$, where (r^x, r^y, ψ) denotes the position and yaw-orientation of the robot base, and q^i denotes the angle of the ith joint of the arm. The control input $u = (u^r, u^\psi, u^1, \dots, u^7)$ corresponds to the forward and angular velocity of the base and to each ith joint velocity. The system follows the dynamics

$$(\dot{r}^x, \dot{r}^y) = (\cos(\psi), \sin(\psi))u^r, \quad \dot{\psi} = u^\psi, \quad \dot{q}^i = u^i, \tag{6}$$

where $i = 1,\dots,7$. The position and rotation matrix of the end-effector are denoted by $p_e(x)$ and $R_e(x)$, respectively. They are defined by the forward kinematics of the system and are computed as a chain of homogeneous transformation matrices $T_i(x) \in SE(3)$, computed using the product of exponentials formula [48]. Given a wiping trajectory $T_t^{\text{ref}} = (p_t^{\text{ref}}, R_t^{\text{ref}})$, we minimize the cost function

$$\inf_u \int_0^T \left(\ell^u(u_t) + \ell^x(x_t, t)\right) \mathrm{d}t, \tag{7}$$

where $T > 0$ is a time-horizon, $\ell^u(u_t) = \|u_t\|^2$ penalizes the control effort, and $\ell^x(x_t, t) = \|p_e(x_t) - p_t^{\text{ref}}\|^2 + \|I - R_e(x_t)^\top R_t^{\text{ref}}\|$ minimizes the end-effector pose tracking error.

We consider two types of obstacle avoidance constraints. First, similarly to [20], self-collisions are avoided by covering the robot with spheres $(p_i(x), r_i)$ and enforcing

$$\|p_i(x) - p_j(x)\| \geq r_i + r_j \tag{8}$$

for all pairs of indices $i \neq j$ that correspond to potential self-intersections given the kinematics of the robot.

Second, we enforce collision avoidance constraints with polytopic obstacles $\mathcal{O}$ detected by the perception stack. Each obstacle is expressed by $q \in \mathbb{N}$ linear inequalities $\mathcal{O} = \{p \in \mathbb{R}^3 : A(p-c) \leq b\}$ with $A \in \mathbb{R}^{q \times 3}$,

$c \in \mathbb{R}^3$, and $b \in \mathbb{R}^q$, where each jth row $A_{j\cdot}$ of A is unit-norm. In practice, we use bounding boxes to enclose obstacles and represent them as polytopes, see Fig. 5. Multiple approaches to enforce collision avoidance constraints with such obstacles exist in the literature, such as decomposing the feasible workspace into convex regions [24], enforcing linearized signed-distance constraints [22], [49], or solving a convex program [50]. We propose a different approach in this work. For any point p, we first compute the index $j^\star(p) = \arg\min_{j=1,\dots,q}\{\gamma_j : \gamma_j \geq 0, \gamma_j A_{j\cdot}^\top (p - c) = b_j\}$ corresponding to the facet of $\mathcal{O}$ that intersects the line from p to c. Then, for any sphere $(p_i(x), r_i)$ covering the robot, we enforce the corresponding r_i-padded halfplane constraint

$$A_{j_i^\star\cdot}^\top (p_i(x) - c) \geq b_{j^\star} + r_i, \tag{9}$$

where $j_i^\star = j^\star(p_i(x))$. This constraint guarantees collision avoidance since $\|A_{j\cdot}\|=1$ and $\mathcal{O}$ is convex. It is differentiable almost everywhere and easy to implement. Using (6)-(9), for an initial state x^0, we obtain the optimal control problem:

$$\textbf{OCP}: \inf_{u\in\mathcal{U}} \int_0^T \ell(x_t, u_t, t)\mathrm{d}t \quad \text{s.t.} \quad x_0 = x^0, \tag{10a}$$

$$\dot{x}_t = f(x_t, u_t), \; c(x_t) \geq 0, \; t \in [0, T]. \tag{10b}$$

All terms in **OCP** are twice differentiable almost everywhere. We discretize **OCP** with an Euler scheme, which results in an optimization problem that can be solved with any off-the-shelf solver. In this work, we leverage the sequential-quadratic-programming shooting method presented in [25]: a computationally efficient method with stable convergence properties, yielding a resilient implementation on the system.

Feedback control: We track the whole-body joint angles using admittance control. This approach enables fine trajectory tracking while wiping with a desired normal force of 10N that meets hardware requirements.

VI. Experimental Evaluation

Offline RL training: We train two wiping policies for crumbs-gathering and spills-cleaning with the same network architecture shown in Fig. 4. For spills wiping, we terminate each episode once there are no visible dirty pixels in the input image. For crumbs-gathering, we terminate once the error in (2) is smaller than 0.02, which corresponds to having all particles within a 15cm radius around the table center. We implement the penalization term $\mathbf{1}\{s_{t_i} \notin \mathcal{T}\}$ by penalizing with a constant term if at least one particle is out of the table, which we found stabilizes training. We train for 50k training and 50 million environment steps with a maximum episode length $N = 20$ and a batch size of 256.

We simulate the SDE using 10^3 particles, discretizing (1) with an Euler-Maruyama scheme with $\Delta t = 0.1$s. Simulating a wipe takes only 5ms with a Python implementation, measured on a computer with a 2.20GHz Intel Xeon CPU. We use $(\alpha, \lambda) = (10^{-2}, 0)$ and $(\alpha, \lambda) = (10^{-2}, 2)$ for the crumbs- and spills-wiping environments. We randomize the initial state by sampling particles from Gaussian distributions, see Fig. 6. We consider 1×1 m tables and a rectangular 30×5 cm wiper W_t^a moving at a constant speed $v = 15\,\frac{\text{cm}}{s}$.

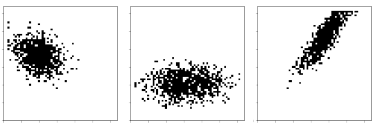

Fig. 6: Initial state distributions for RL training.

Simulation results: We validate our RL policies in the environments with initial distributions shown in Fig. 6, and compare them with a baseline that always wipes to the center of the table with an orientation rotating around the table in increments of $\pi/4$. We also compared with a method that wipes along the direction of largest covariance, which we found had worse performance due to the multimodal particle distribution after the initial wipe. We present results for aggregated 10^3 rollouts in Fig. 7. Error bars correspond to ± 1 standard deviation. Results demonstrate that our RL policies wipe significantly faster than the baseline.

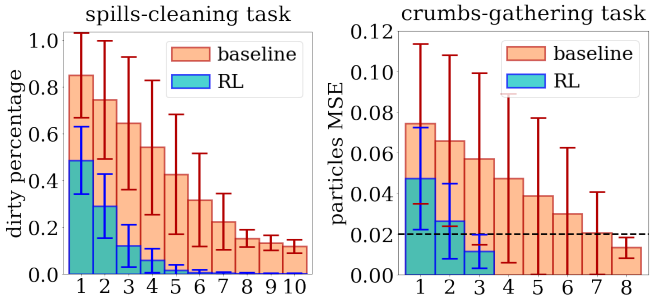

Fig. 7: Comparisons between the RL wiping policy and the baseline. The x-axis denotes the number i of executed wipes, see Section IV.

We test these two policies on environments with initial states sampled from a mixture of Gaussians representing multiple unclean table areas and show results in Fig. 8. We observe that despite having never encountered these states at training time, the RL policies generalize to these observations. This is most likely due to the inductive bias from the CNN policy architecture. Remarkably, the policy never wipes out of the table in our experiments.

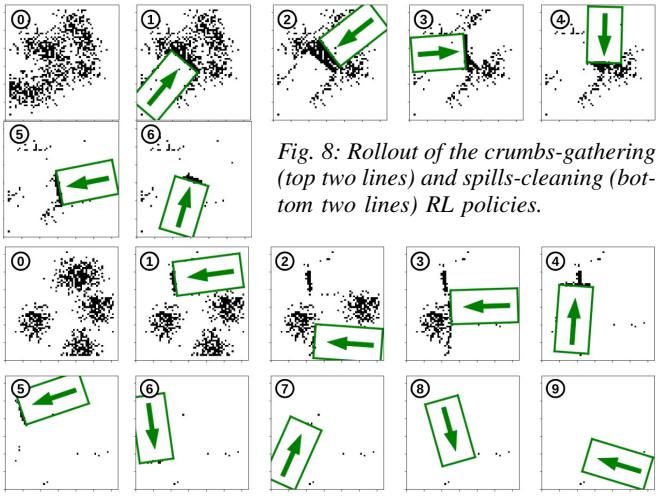

Fig. 8: Rollout of the crumbs-gathering (top two lines) and spills-cleaning (bottom two lines) RL policies.

Hardware setup: We consider crumbs-gathering and spill-wiping scenarios on a table with two chairs that limit the motion of the robot. For crumbs-gathering tasks, we compute a wiping trajectory to a bin once all crumbs are closely together. For spills-wiping, we terminate the task once no

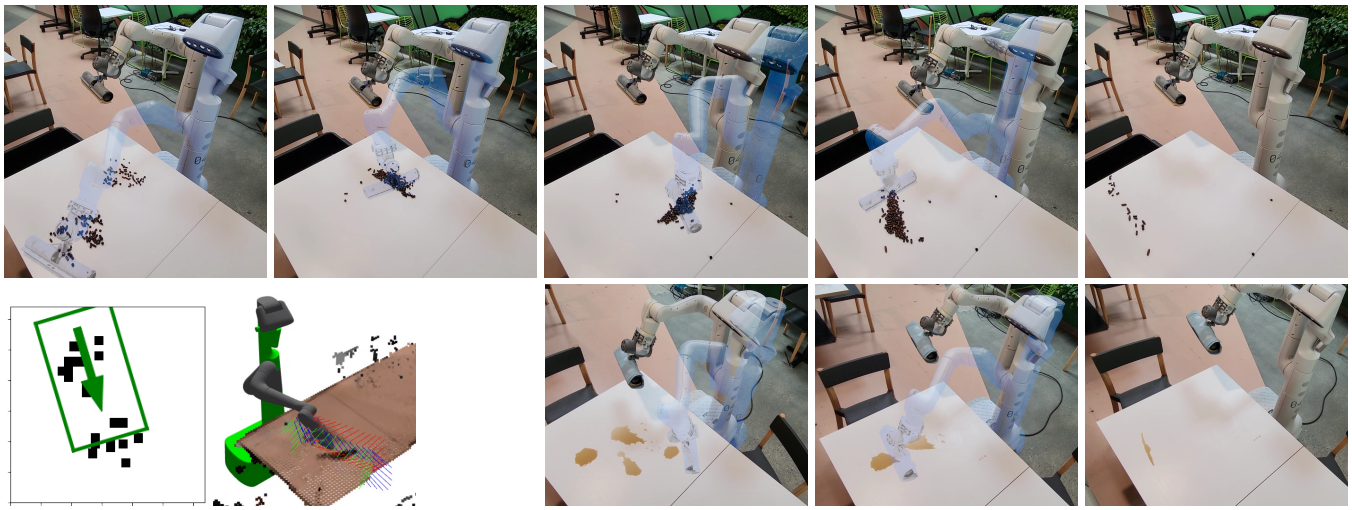

Fig. 9: Results on the assistive robot from Everyday Robots on crumbs-gathering (top and bottom left) and spills-wiping tasks (bottom).

particles are detected by the perception module. To obtain inputs for the RL policy, we use color filtering from images to obtain a mask o of resolution 64×64 pixels with approximate locations of dirty particles on the table. Since this outputs a sparse representation of perceived spill and crumbs, we inflate the detected particles by two pixels before passing this input to the RL policy. Inference time of the network is 7 ms. Wiping actions are then converted to a desired pose trajectory $(p_t^{\text{ref}}, R_t^{\text{ref}})$ on the table passing from the current tool pose, the wiping poses, and back to the initial tool pose. To wipe effectively, we minimize the yaw and roll orientation error but do not consider the pitch angle of the end-effector with respect to the table for trajectory optimization. Computing a whole-body collision-free trajectory takes about 500 ms with zero-controls as the initial guess. Finally, the admittance controller tracks the joint trajectory while applying a 10 N normal force with the table, meeting hardware requirements and yielding satisfactory wiping results.

Hardware results: We present results in Fig. 9 and in the supplementary video. We observe that the RL policy generalizes well, thanks to the design of the observation space that minimizes the simulation to real gap. In all scenarios, the wipes planned by the RL policy point in the direction of the center of the table to avoid moving crumbs and spills out of the table. The trajectory optimizer computes whole-body trajectories that successfully avoid obstacles with chairs and the table, while operating in close proximity with obstacles.

To validate our simulator, we use observations from the system to initialize the simulator and predict the evolution of perceived particles for a planned wipe. Then, we compare the result with perceived particles after executing the wipe on the system. In Fig. 10, we present results for the first wipe of the crumbs-gathering scenario shown in Fig. 9. We observe that predictions are reasonable and correctly predict the general trend of crumbs particles. This result explains the success of hardware experiments: despite the imperfection of the simulator, its accuracy is sufficient to train RL policies that yield useful wipes to efficiently clean the table. An extensive quantitative validation of the simulator is beyond the scope of this work and left for future research.

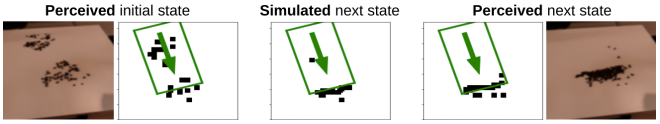

Fig. 10: Initial image and planned wipe (left), prediction after wipe from simulator (middle), and final observation after wiping (right).

VII. Conclusion

We present a new approach to enable a mobile manipulator to wipe tables and clean spills and crumbs. The key consists of decomposing the task: we first train an RL policy to compute high-level wiping waypoints, and subsequently compute whole-body trajectories for the robot to safely execute these wipes. Our RL policy is trained entirely in simulation using an analytic SDE model of spill and crumbs dynamics and thus does not require collecting data on the robotic platform. We show that our approach outperforms wiping baselines and demonstrate our framework in hardware experiments.

Future work: This work opens exciting avenues of future research. First, collected data from the system could be used to train a perception module to better detect and distinguish between different types of dirty particles [26]. Second, one could plan more complex wiping motions by combining different wiping primitives. Third, the parameters of the SDE in (1) could be learned, e.g. using neural SDEs [51], [52]. They could potentially be inferred online by observing the outcome of wipes, e.g. using meta-learning approaches [40]. Finally, the SDE in (1) and state representation could allow inferring and reasoning about the time-varying wiper spill-absorption properties: Since the variable s^z models cleaned particles, one could keep track of the amount of wiped particles and incorporate this information for planning, perhaps by modeling the absorption parameter λ as a stochastic process λ_t that depends on the amount of wiped particles.

Acknowledgements

We would like to thank Benjie Holson, Jake Lee, April Zitkovich, and Linda Luu for their help and support with experiments.

REFERENCES

[1] K. Kim, A. K. Mishra, R. Limosani, M. Scafuro, N. Cauli, J. Santos-Victor, B. Mazzolai, and F. Cavallo, "Control strategies for cleaning robots in domestic applications: A comprehensive review," *Int. Journal of Advanced Robotic Systems*, vol. 16, no. 4, pp. 1–21, 2019.

[2] M. Bajracharya, J. Borders, D. Helmick, T. Kollar, M. Laskey, J. Leichty, J. Ma, U. Nagarajan, A. Ochiai, J. Petersen, K. Shankar, K. Stone, and Y. Takaoka, "A mobile manipulation system for one-shot teaching of complex tasks in homes," in *Proc. IEEE Conf. on Robotics and Automation*, 2020.

[3] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Conf. on Robot Learning*, 2018.

[4] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.

[5] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "RL-CycleGAN: Reinforcement learning aware simulation-to-real," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2020.

[6] A. Ghadirzadeh, X. Chen, P. Poklukar, C. Finn, M. Bjorkman, and D. Kragicm, "Bayesian meta-learning for few-shot policy adaptation across robotic platforms," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2021.

[7] F. B. Hanson, *Applied Stochastic Processes and Control for Jump-Diffusions*. SIAM, 2007.

[8] B. Øksendal and A. Sulem, *Applied Stochastic Control of Jump Diffusions*. Springer Berlin Heidelberg, 2007.

[9] E. A. Theodorou and E. Todorov, "Stochastic optimal control for nonlinear Markov jump diffusion processes," in *American Control Conference*, 2012.

[10] T. Lew, R. Bonall, and M. Pavone, "Sample average approximation for stochastic programming with equality constraints," 2022, available at https://arxiv.org/abs/2206.09963.

[11] R. Martin-Martin, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2019.

[12] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, 2020.

[13] F. Xia, C. Li, R. Martin-Martin, O. Litany, A. Toshev, and S. Savarese, "ReLMoGen: Integrating motion generation in reinforcement learning for mobile manipulation," in *Proc. IEEE Conf. on Robotics and Automation*, 2021.

[14] J. Wu, X. Sun, A. Zeng, S. Song, S. Rusinkiewicz, and T. Funkhouser, "Learning pneumatic non-prehensile manipulation with a mobile blower," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2022.

[15] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin, "Combining optimal control and learning for visual navigation in novel environments," in *Conf. on Robot Learning*, 2020.

[16] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body MPC for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687–3694, 2019.

[17] M. V. Minniti, R. Grandia, K. Fäh, F. Farshidian, and M. Hutter, "Model predictive robot-environment interaction control for mobile manipulation tasks," in *Proc. IEEE Conf. on Robotics and Automation*, 2021.

[18] C. D. Bellicoso, K. Kramer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, "High-frequency nonlinear model predictive control of a manipulator," in *Proc. IEEE Conf. on Robotics and Automation*, 2019.

[19] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified MPC framework for whole-body dynamic locomotion and manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4688–4695, 2021.

[20] J.-R. Chiu, J.-P. Sleiman, M. Mittal, F. Farshidian, and M. Hutter, "A collision-free MPC for whole-body dynamic locomotion and manipulation," in *Proc. IEEE Conf. on Robotics and Automation*, 2022.

[21] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *Int. Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[22] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *Int. Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[23] J. Tordesillas and J. P. How, "MADER: Trajectory planner in multi-agent and dynamic environments," *IEEE Transactions on Robotics*, vol. 38, no. 1, 2022.

[24] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," 2022, available at https://arxiv.org/abs/2205.04422.

[25] S. Singh, J.-J. Slotine, and V. Sindhwani, "Optimizing trajectories with closed-loop dynamic SQP," in *Proc. IEEE Conf. on Robotics and Automation*, 2022.

[26] J. Yin, K. G. S. Apuroop, Y. K. Tamilselvam, R. E. Mohan, B. Ramalingam, and A. Le, "Table cleaning task by human support robot using deep learning technique," *Sensors*, vol. 20, no. 6, pp. 1698–1717, 2020.

[27] J. Hess, J. Sturm, and W. Burgard, "Learning the state transition model to efficiently clean surfaces with mobile manipulation robots," in *Proc. of the Workshop on Manipulation under Uncertainty, IEEE Conf. on Robotics and Automation*, 2011.

[28] D. Leidner, W. Bejjani, A. Albu-Schäffer, and M. Beetz, "Robotic agents representing, reasoning, and executing wiping tasks for daily household chores," in *Proc. Int. Conf. on Autonomous Agents and Multiagent Systems*, 2011.

[29] S. Elliott and M. Cakmak, "Robotic cleaning through dirt rearrangement planning with learned transition models," in *Proc. IEEE Conf. on Robotics and Automation*, 2018.

[30] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *Proc. IEEE Conf. on Robotics and Automation*, 2017.

[31] J. Kim, N. Cauli, P. Vicente, B. Damas, F. Cavallo, and J. Santos-Victor, ""iCub, clean the table!" a robot learning from demonstration approach using deep neural networks," in *Proc. IEEE Int. Conf. on Autonomous Robot Systems and Competitions*, 2018.

[32] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," in *Proc. IEEE Conf. on Robotics and Automation*, 2018.

[33] A. Gams, T. Petrič, M. Do, B. Nemec, J. Morimoto, T. Asfour, and A. Ude, "Adaptation and coaching of periodic motion primitives through physical and visual interaction," *Robotics and Autonomous Systems*, vol. 75, pp. 340–351, 2016.

[34] S. Elliott, Z. Xu, and M. Cakmak, "Learning generalizable surface cleaning actions from demonstration," in *Proc. IEEE Int. Symp. on Robot and Human Interactive Communication (RO-MAN)*, 2017.

[35] N. Cauli, P. Vicente, J. Kim, B. Damas, A. Bernardino, F. Cavallo, and J. Santos-Victor, "Autonomous table-cleaning from kinesthetic demonstrations using deep learning," in *Proc. IEEE Int. Conf. on Development and Learning and Epigenetic Robotics*, 2018.

[36] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and where pathways for robotic manipulation," in *Conf. on Robot Learning*, 2021.

[37] D. Applebaum, *Lévy Processes and Stochastic Calculus*, 2nd ed. Cambridge University Press, 2009.

[38] H. J. H.J. Terry Suh and R. Tedrake, "The surprising effectiveness of linear models for visual foresight in object pile manipulation," in *Workshop on Algorithmic Foundations of Robotics*, 2020.

[39] C. Jiang, C. Schroeder, J. Teran, A. Stomakhin, and A. Selle, "The material point method for simulating continuum materials," in *ACM SIGGRAPH 2016 Courses*, 2016.

[40] T. Lew, A. Sharma, J. Harrison, A. Bylard, and M. Pavone, "Safe active dynamics learning and control: A sequential exploration-exploitation framework," *IEEE Transactions on Robotics*, 2022.

[41] E. Altman, *Constrained Markov Decision Processes*, 1st ed. Chapman & Hall/CRC, 1999.

[42] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A lyapunov-based approach to safe reinforcement learning," in *Conf. on Neural Information Processing Systems*, 2018.

[43] Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Ghavamzadeh, "Lyapunov-based safe policy optimization for continuous control," in *Conf. on Robot Learning*, 2020.

[44] G. Thomas, Y. Luo, and T. Ma, "Safe reinforcement learning by imagining the near future," in *Conf. on Neural Information Processing Systems*, 2021.

[45] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proc. AAAI Conf. on Artificial Intelligence*, 2018.

[46] S. Gu, L. Yang, Y. Du, G. Chen, F. Walter, J. Wang, Y. Yang, and A. Knoll, "A review of safe reinforcement learning: Methods, theory and applications," 2022, available at https://arxiv.org/abs/2205.10330.

[47] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Int. Conf. on Machine Learning*, 2018.

[48] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge University Press, 2017.

[49] T. Lew, R. Bonalli, and M. Pavone, "Chance-constrained sequential convex programming for robust trajectory optimization," in *European Control Conference*, 2020.

[50] K. Tracy, T. A. Howell, and Z. Manchester, "DiffPills: Differentiable collision detection for capsules and padded polygons," 2022, available at https://arxiv.org/abs/2207.00669.

[51] P. Kidger, J. Foster, X. Li, H. Oberhauser, and T. Lyons, "Neural SDEs as infinite-dimensional GANs," in *Int. Conf. on Machine Learning*, 2021.

[52] A. R. Benson and J. Jia, "Neural jump stochastic differential equations," in *Conf. on Neural Information Processing Systems*, 2019.