Push-Wiper: Toward General-Purpose Robotic Cleaning across Varied Stains and Surfaces with Segmented Pushing Trajectories

Renhao Lu[1,2,†], Mingxin Wang[1,†], Chenyang Cao[3], Yang Yang[1], Guoping Pan[1,2], Kangkang Dong[1], Yi Cheng[2], Houde Liu[1,2,*]

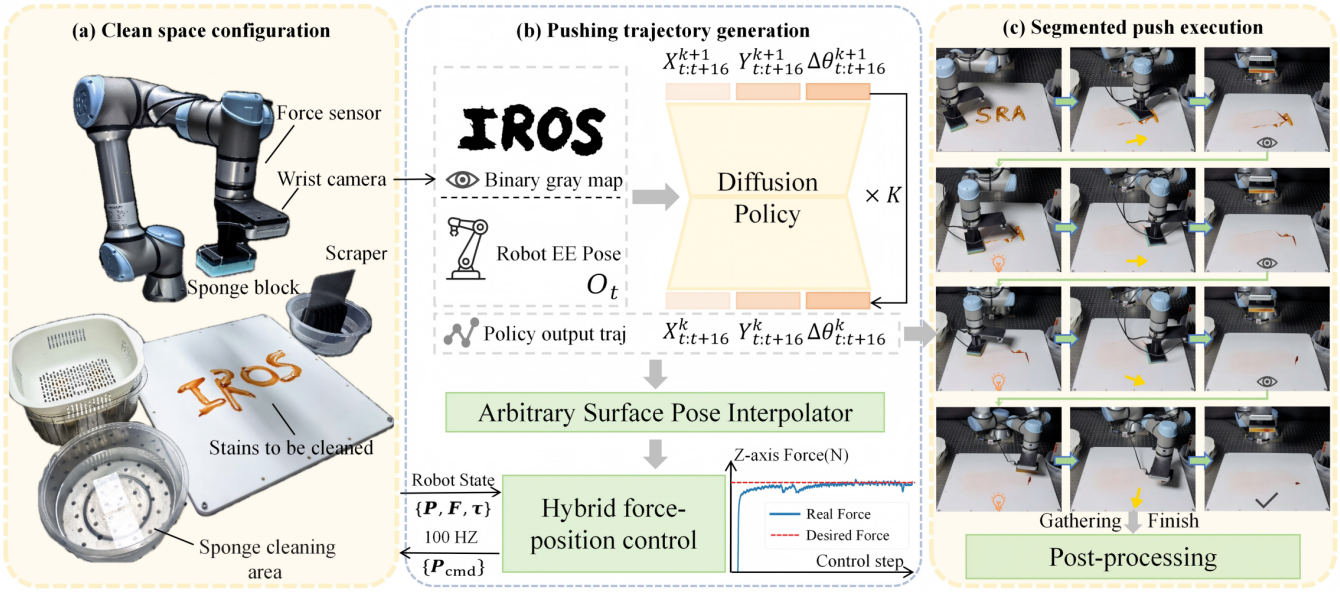


Fig. 1: **Overview of the Push-Wiper framework.** (a) Experimental setup and tools configuration. (b) Trajectory generation: From one observation, Push-Wiper uses Diffusion Policy to produce 3D pushing actions $(x, y, \Delta\theta)$, which are converted into a smooth executable trajectory by our Arbitrary Surface Pose Interpolator and Hybrid force–position controller. (c) Through iterative *Perception–Planning–Execution* cycles, the robot gathers stains into a small region, after which Post-processing removes the residues to complete cleaning.

***Abstract*— Viscous stains, characterized by high viscosity and complex rheological properties, remain a major challenge for robotic surface cleaning. Conventional wiping often spreads the stain and scrubbing provides stronger friction but risks damaging the surface. In this paper, we propose Push-Wiper, a framework that reformulates viscous stain cleaning as an aggregation problem. Push-Wiper employs a sponge to progressively gather stains through segmented pushing trajectories, followed by a post-processing phase that detaches the aggregated material and enables sponge self-cleaning. We adopt a stepwise strategy for stain gathering and leverage Diffusion Policy to generate adaptive pushing action sequences and execute them via our Arbitrary Surface Pose Interpolator (ASPI) together with a Hybrid force–position controller, allowing the method to generalize to stains with diverse spatial distributions. Push-Wiper achieves a cleaning score (CS), defined as the percentage of stain area removed, up to 130% higher than baseline methods. Without additional training, Push-Wiper also transfers in a zero-shot manner to solid residues, liquid spills, unseen viscous stains and curved surfaces with varying geometries. Our experiments demonstrate the cleaning effectiveness of Push-Wiper and its strong generalization ability. The project website is available at `https://push-wiper.github.io/`.**

I. Introduction

Cleaning robots are now widely deployed in domestic, commercial, and industrial settings, with rapid advances in capability and scope in recent years [1]. Similarly, the task of wiping has been automated for cleaning regular surfaces like floors [2], windows [3], and tables [4]. Although conventional robotic cleaning methods, such as sweeping and wiping, can effectively address solid residues and liquid spills in tabletop tasks, viscous stains continue to pose a

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China.
[2]Z-Lab, Zerith Robotics, Shenzhen 518055, China.
[3]University of Toronto, Toronto M5S 1A1, Canada.
*Corresponding author: Houde Liu (liu.hd@sz.tsinghua.edu.cn).
This work was supported by the Shenzhen Science and Technology Program (Grant No. RCJC20210706091946001) and the Shenzhen Science and Technology Program (Grant No. ZDCY20250901104207008). († indicates equal contribution).

significant challenge [5].

Existing in a semi-solid state, viscous stains exhibit high viscosity and complex rheological properties [6]. Consequently, they are hard to remove and often smear or spread, worsening contamination. Effective cleaning therefore requires applying a controlled normal force and generating sufficient tangential friction to shear the material off the surface [7]. Unlike dishwashing, where abundant rinsing is feasible [8], tabletop settings restrict liquid use, further complicating cleaning.

Wiping and scrubbing are two common motions in robotic surface cleaning [5]. Scrubbing, which employs tools like brushes in reciprocating motions, is effective for removing viscous stains. However, it poses a higher risk of surface abrasion and thus demands precise force control. Wiping, on the other hand, is highly effective for absorbable contaminants like liquid spills, but it is prone to causing secondary contamination by spreading viscous substances. Additionally, in real-world scenarios, tabletop stains are not uniform and can manifest in various forms, such as solid debris, liquid spills, and viscous substances [9]. Therefore, a cleaning robot can hardly rely on a single strategy or tool for effective cleaning. This often requires the combination of different approaches [10], [11], which in turn leads to greater complexity in both design and implementation.

In this work, we propose Push-Wiper, a learning-based framework specifically designed to tackle the complex rheological properties of viscous stains (Fig. 1). Push-Wiper fundamentally reformulates viscous stain cleaning as a topological aggregation problem. To bypass the intractable analytical modeling of fluid deformation under contact, we propose an "aggregate-then-finish" paradigm. In the gathering phase, Push-Wiper operates episodically. It utilizes Diffusion Policy [12] as a low-frequency action generator, inferring segmented low-dimensional pushing actions strictly from abstract binary observations. This deliberate abstraction forces the policy to learn multimodal aggregation behaviors independent of complex visual textures or 3D surface geometries. To execute these local aggregation steps, Push-Wiper employs an Arbitrary Surface Pose Interpolator (ASPI) combined with a hybrid force–position controller. This architecture strictly decouples 2D topological planning from 3D geometric execution, directly mapping the low-dimensional action sequence to a high-dimensional end-effector pose trajectory compatible with arbitrary surfaces. Finally, the post-processing phase introduces motion primitives to detach aggregated residue and self-clean the sponge for reuse. This decoupled design improves cleaning performance and enables robust zero-shot generalization to unseen curved surfaces and diverse stain categories without additional data or retraining.

In summary, our main contributions are as follows:

- **A Novel Paradigm for Viscous Stain Cleaning:** We cast viscous stain removal as a state-aggregation problem and propose an "aggregate-then-finish" strategy that overcomes the limitations of conventional wiping and scrubbing.
- **A Decoupled Visuomotor Architecture:** We propose Push-Wiper, which couples a low-frequency Diffusion Policy for 3D action generation with ASPI and hybrid force–position control for 6D trajectory. This decoupling isolates the policy from 3D surface variations, reducing problem complexity.
- **Strong Zero-Shot Generalization:** We demonstrate that, by abstracting the task into geometric topologies rather than complex visual textures or 3D surface geometries, our system generalizes zero-shot to unseen curved surfaces, solid residues, liquid spills, and novel viscous stains. Experiments show Push-Wiper achieves near-complete cleaning and improves cleaning score by up to 130% over baselines.

II. Related Work

In this section, we review existing methods for robotic surface cleaning. Based on their approach to generating action trajectories, these methods can be broadly categorized into two groups: classical and learning-based methods.

Classical methods. Classical robotic surface cleaning methods typically rely on predefined trajectories executed through position, force, or compliant control. Hess et al. [13] and Wang et al. [14] perform coverage path planning and track the planned wiping motions using position control. To handle physical interaction with the environment, Ortenzi et al. [15] incorporate contact constraints for robotic whiteboard wiping, while Leidner et al. [16] achieve compliant surface cleaning through impedance control [17]. For viscous stains, Harmatz et al. [4] and Kowalewski et al. [5] adopt hybrid force–position control with soft robotic arms. However, they still have some limitations: the former requires human intervention, while the latter still leaves substantial residues.

Learning-based methods. Learning-based cleaning methods mainly follow two paradigms: reinforcement learning (RL) and imitation learning (IL). RL formulates cleaning as reward-driven policy learning. Lew et al. [11] design rewards for liquid-spill cleaning to learn high-level wiping strategies, but their policy relies only on position control despite the importance of force feedback. Mart'ın-Mart'ın et al. [18] combine PPO with variable impedance control for wiping, yet may suffer performance degradation during sim-to-real transfer. IL learns cleaning strategies from expert demonstrations. Tsuji et al. [19] incorporate real-time force-torque feedback and pre-trained object representations into IL, enabling generalization to plane-height variations. Oishi et al. [20] propose a motion generation model that modulates velocity and contact force during command-conditioned whiteboard wiping. Recently, diffusion models have been increasingly applied to robotics [21], with several works [22]–[24] integrating force information into diffusion-based wiping policies. However, these methods mainly address simple mark-wiping tasks on a single surface type, such as whiteboards, and do not tackle viscous stain cleaning in diverse environments.

Summary. Classical methods are stable for predefined surface cleaning tasks but rely on precise models and hand-tuned parameters, limiting their portability across platforms

and environments. Learning-based methods improve adaptability within specific scenarios, yet generalization to new conditions often requires redesign or retraining. Currently, there is no single strategy that can effectively handle multiple types of stains on arbitrary curved surfaces. To the best of our knowledge, no learning-based framework has been specifically developed for the challenging task of cleaning viscous stains.

III. Push-Wiper Framework

A. Problem decomposition

The objective of our work is to completely remove viscous stains from the surface. We simplify the surface D into a discrete $M \times N$ grid, where the state of each cell is denoted by $D_{i,j} \in \{0, 1\}$. 0 and 1 represent a dirty state and a clean state, respectively. Our objective is formulated as follows:

$$\begin{aligned} \max_{\{D_{i,j}\}} \quad & \sum_{i=1}^{M} \sum_{j=1}^{N} D_{i,j} \\ \text{s.t.} \quad & D_{i,j} \in \{0, 1\}. \end{aligned} \tag{1}$$

For broad compatibility across robotic systems, our framework requires only basic hardware: a manipulator with a wrist camera and a simple tool, such as a sponge.

Directly wiping viscous stains often exacerbates contamination due to their complex rheology. Inspired by human strategies, Push-Wiper (Fig. 1) decouples the task into a *gathering phase* and a *post-processing phase*. Let $\mathcal{S}_t$ denote the set of stain pixels at step t. We explicitly formulate stain aggregation as minimizing its maximum spatial diameter $\mathcal{D}(\mathcal{S}_t)$ and connectivity fragmentation $K(\mathcal{S}_t)$ via segmented pushing trajectories τ:

$$\min_{\tau} \quad \mathcal{D}(\mathcal{S}_{t+1}) + \lambda K(\mathcal{S}_{t+1}), \tag{2}$$

where λ balances the spatial spread and the number of disconnected components. Analytically predicting such fluid deformation under contact requires precise prior knowledge of hidden rheological parameters, making classical planning intractable [25], [26]. This physically motivates our data-driven approach. Using binary maps and robot states as observations, the gathering phase progressively contracts the stain to a compact state (i.e. $\mathcal{D} < \epsilon, K \to 1$). Finally, the post-processing phase executes predefined primitives to remove the gathered residue, achieving complete cleaning.

B. Gathering Phase

Action generation. To align our data-driven approach with the aggregation objective in Eq. (2), we deliberately abstract the visual observation into a texture-less binary stain map $\mathbf{M}_t$ (stain $= 0$, clean $= 1$). This geometric abstraction eliminates the policy's reliance on visual appearance, forcing the robot to learn pushing strategies purely from the stain's spatial topology. Because a scattered stain distribution can be aggregated from multiple valid directions, we adopt Diffusion Policy (DP) [12] to capture this inherent multimodal distribution of expert demonstrations.

Standard DP operates in a high-frequency receding-horizon manner with continuous visual feedback, which cannot be reliably maintained under our minimal sensing setting: the wrist camera loses view during contact, and an external camera view is often occluded by the arm. Conversely, planning the entire gathering phase from a single initial observation is also ineffective, as the severe redistribution of viscous stains during pushing quickly invalidates the initial plan. To explicitly address these dual challenges and encode the aggregation behavior, we curate our expert demonstrations as discrete, segmented pushing strokes rather than continuous, end-to-end cleaning episodes. We view each pushing trajectory as a local optimization step that aims to decrease the aggregation objective $J_t = \mathcal{D}(\mathcal{S}_t) + \lambda K(\mathcal{S}_t)$ in Eq. (2). Consequently, we deploy DP as a low-frequency, macro-level planner: re-perceive, plan, execute open-loop, and re-perceive.

The iterative gathering process is shown in Fig. 1(b) and (c). At the beginning of step t, the observation $\mathbf{O}_t = \{\mathbf{M}_t, \mathbf{p}_{cap}\}$ (where $\mathbf{p}_{cap}$ is the fixed capture pose) is fed into the policy π to generate an n-step action sequence $\mathbf{A}_t$. Because of our segmented training paradigm, $\mathbf{A}_t$ does not merely represent a short-term receding horizon, but strictly encodes a complete pushing trajectory. This temporal discretization transforms DP into a local aggregation operator: at each macro-step, it infers a full stroke to progressively reduce the stain's spatial diameter.

Crucially, we restrict the action space to $a = (x_b, y_b, \Delta\theta)$, representing 2D planar translations and yaw rotations relative to $\mathbf{p}_{cap}$. Rather than outputting arbitrary 6D trajectories, this low-dimensional space strictly regularizes the policy to generate planar aggregating motions that push the boundaries of $\mathbf{M}_t$ inward. By decoupling this 2D topological planning from the 3D surface geometry, ASPI can independently reconstruct the 6-DoF end-effector pose. This design inherently grounds the learned actions in our aggregation formulation, drastically reducing the search space and enabling zero-shot generalization across unseen curved surfaces.

Trajectory generation. Algorithm 1 outlines ASPI. While mapping low-dimensional trajectories onto 3D surfaces is an effective approach for robotic operations [27], we uniquely deploy it here to explicitly decouple 2D topological inference

Algorithm 1 ArbitrarySurfacePoseInterpolator($\mathbf{A}_t$, $\mathbf{p}_{cap}$, f)

$\mathbf{P}_{seq} \leftarrow \{\}$
for $\mathbf{a} \in \mathbf{A}_t$ **do**
  $x_m, y_m, \Delta\theta \leftarrow PointInSurface(\mathbf{a}, \mathbf{p}_{cap})$
  $z_m, \mathbf{n}_m \leftarrow GetSurfaceNormal(x_m, y_m)$
  $\mathbf{p}_{base} \leftarrow GeneratePose(x_m, y_m, z_m, -\mathbf{n}_m, \Delta\theta)$
  $\mathbf{P}_{seq} \leftarrow \mathbf{P}_{seq} \cup \{\mathbf{p}_{base}\}$
$\mathbf{t}_{seq}, \phi_{seq} \leftarrow \mathbf{P}_{seq}$
$\mathbf{t}_{traj} \leftarrow BslineAndTrapVel(\mathbf{t}_{seq}, f)$
$\phi_{traj} \leftarrow Slerp(\phi_{seq}, f)$
$\mathbf{P}_{traj} \leftarrow Hstack(\mathbf{t}_{traj}, \phi_{traj})$
Return $\mathbf{P}_{traj}$

from 3D geometric execution. For each 3D planar action $\mathbf{a} \in \mathbf{A}_t$, ASPI projects it onto the 3D surface to extract the corresponding point (x_m, y_m, z_m) and normal vector $\mathbf{n}_m$ (lines 1-6). To ensure effective gathering of viscous stains and maintain stable physical contact, we enforce a strict geometric pose constraint on the end-effector: the z-axis of the TCP is strictly aligned with $-\mathbf{n}_m$, and the model predicts an additional yaw increment $\Delta\theta$, which is superimposed on the reference pose p_{cap}. Furthermore, because our policy outputs sparse macro-waypoints, direct execution would induce acceleration transients that destabilize the subsequent hybrid force-position controller. Thus, ASPI applies B-spline fitting, trapezoidal velocity time parameterization, and spherical linear interpolation to synthesize a 6D trajectory $\mathbf{P}_{traj}$ at f Hz (lines 7-10). Crucially, rather than requiring the intractable collection of expert demonstrations across diverse curved geometries, this deterministic 2D-to-3D bridge explicitly shields the policy from the burden of modeling complex 3D surface variations. By learning topological aggregation exclusively from planar data, we radically reduce the problem complexity, which inherently enables our zero-shot generalization across unseen curved surfaces.

Trajectory execution. To push viscous stains while executing $\mathbf{P}_{traj}$, Push-Wiper maintains a prescribed contact force to keep the sponge in close contact with the surface. In our setting, successful execution primarily depends on maintaining sufficient normal contact during motion. We therefore handle force feedback at the execution layer rather than using it as a policy input. We therefore keep force feedback in the execution layer and adopt a hybrid force–position controller that decouples force regulation from motion tracking. Specifically, Push-Wiper uses an admittance controller [28] to regulate the normal contact force to a constant setpoint F_z^{des}. Since each pose in $\mathbf{P}_{traj}$ aligns the TCP z-axis with the surface normal, this is equivalent to regulating the TCP z-axis force component to F_z^{des}. The admittance controller is as follows:

$$m\,\ddot{\delta z} + b\,\dot{\delta z} + k\,\delta z = F_z^{des} - F_z, \tag{3}$$

where m, b, k denote the inertia, damping and stiffness matrices. F_z is the feedback of the force sensor and δz is the correction value of the admittance controller. The remaining translational axes and end-effector orientation follow $\mathbf{P}_{traj}$ under position control. In the TCP frame, the x–y tracking error $\Delta\mathbf{s}^E_{\text{track}}$ and admittance increment $\Delta\mathbf{s}^E_{\text{adm}}$ are:

$$\Delta\mathbf{s}^E_{\text{track}} = \left[(\mathbf{R}^B_E)^\top\left(\mathbf{s}^B_{\text{traj}} - \mathbf{s}^B_E\right)\right]_{x,y},\ \Delta\mathbf{s}^E_{\text{adm}} = \begin{bmatrix}0\\0\\\delta z\end{bmatrix}, \tag{4}$$

where $\mathbf{R}^B_E$ is the rotation matrix from the TCP frame to the base frame. $\mathbf{s}^B_{\text{traj}}$ and $\mathbf{s}^B_{\text{E}}$ denote the desired trajectory position and the measured end-effector position in the base frame, respectively. The final pose command sent to the manipulator is:

$$\mathbf{s}^B_{\text{cmd}} = \mathbf{s}^B_E + \mathbf{R}^B_E\left(\Delta\mathbf{s}^E_{\text{track}} + \Delta\mathbf{s}^E_{\text{adm}}\right),\ \mathbf{R}^B_{\text{cmd}} = \mathbf{R}^B_{\text{traj}}, \tag{5}$$

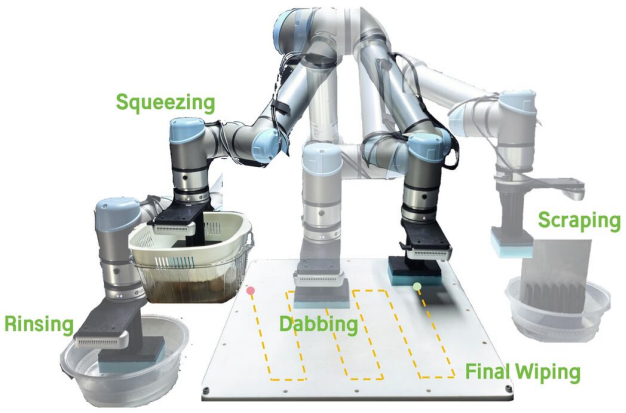


Fig. 2: **Five motion primitives in the post-processing phase.** The yellow dashed line indicates the full-cover path in the *final wiping*.

where $\mathbf{R}^B_{\text{traj}}$ is the orientation in expected trajectory point.

To reduce residual stains, after each pushing trajectory the manipulator performs a predefined scraping motion to clean the sponge. During gathering, it repeats pushing until the stain area drops below q_{th}.

C. Post-processing Phase

The post-processing phase removes the gathered residue and completes the cleaning process. As shown in Fig. 2, after the gathering phase aggregates the stain into a compact region, the robot executes a small set of predefined motion primitives with the same hybrid force–position controller. These primitives include dabbing to lift the residue, scraping, rinsing, and squeezing to self-clean the sponge and regulate moisture, followed by a final full-coverage wipe. This phase improves cleaning completeness and supports repeated sponge use within the framework.

IV. Experiments

Our experiments include: (i) comparative evaluation of Push-Wiper (without post-processing) against two baselines on two representative viscous stains, (ii) validation of its generalization on unseen objects and arbitrary curved surfaces, and (iii) evaluation of the effectiveness of post-processing.

A. Experimental Setup

Platform. As shown in Fig. 1(a), we use a UR7e robot with a KWR75B six-axis force/torque sensor for hybrid force–position control and an Intel RealSense D435 wrist camera for capturing stain images. An $11\,\text{cm} \times 7\,\text{cm}$ sponge block is mounted via a 3D-printed adapter, requiring no specialized mechanisms. All devices are connected to a workstation with an Intel Core i7-14700F CPU and NVIDIA RTX 4060Ti GPU for data collection and policy evaluation.

Tasks. We categorize viscous stains into two types:

- **Simple** type: the surface contains only one connected stain region, and the stain pixels account for less than $20\,\%$ of the entire image, such as simple shapes resembling numbers or letters.

- **Complex** type: the surface contains two or more connected stain regions, or the stain pixels account for more than 20 % of the image, such as multiple numbers distributed in a scattered manner.

Metrics. We evaluate our system on two representative viscous stains with distinct physical properties: ketchup and peanut butter. Ketchup exhibits an apparent viscosity of $1000\,\mathrm{mPa}\cdot\mathrm{s}$ to $1500\,\mathrm{mPa}\cdot\mathrm{s}$ at a shear rate of $10\,\mathrm{s}^{-1}$ [29], whereas peanut butter reaches $29\,452\,\mathrm{mPa}\cdot\mathrm{s}$ under the same condition [30]. This near order-of-magnitude gap implies much higher adhesion for peanut butter, making it more challenging to clean.

To quantitatively compare the cleaning performance under these challenging conditions, we define the Cleaning Score (CS). Specifically, we extract a denoised stain mask using an area-based detector that fuses HSV/Lab/grayscale cues with simple morphology and connected-component filtering. Let N_{before} and N_{after} denote the total number of stain pixels before and after each cleaning, respectively. The Cleaning Score (CS) is then defined as the percentage reduction in stain pixels after cleaning:

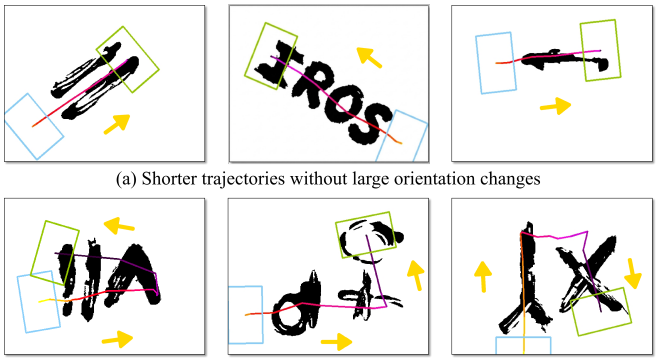
$$\text{CS} = \left(1 - \frac{N_{\text{after}}}{N_{\text{before}}}\right) \times 100 \tag{6}$$

B. Baselines and Implementation

Baselines. To isolate the benefit of the proposed aggregation-first paradigm under a controlled and reproducible setup, we implement two strong baseline strategies within the same perception and execution stack. This design minimizes confounding factors from implementation differences, while covering two common alternatives: coverage-style wiping and one-shot pushing.

- **Full-Cover (FC):** A fixed full-coverage sweeping trajectory (e.g., arched or boustrophedon-like) that spans the entire cleaning area is executed repeatedly.
- **PushAll-Onetime(PO):** To test whether our segmented pushing is necessary, we construct a baseline that follows the same episodic *re-observe–plan–execute* loop as Push-Wiper, but uses a fundamentally different *per-iteration plan*. At each replanning step, PO predicts a *single global* continuous pushing trajectory intended to traverse and sweep through the entire current stain region in one stroke, rather than producing short strokes that progressively aggregate the stain. Consequently, PO typically yields long paths and large orientation changes for large or spatially scattered stains. As shown in Fig. 3, we use a lightweight 2D trajectory synthesis tool implemented in Pygame to generate supervisory *global sweep* trajectories on the same binary stain-map domain derived from real-world demonstrations. This synthesis is used solely for mask-space *planning supervision* rather than contact-rich execution simulation, enabling a controlled comparison of "global sweep" versus our aggregation-first segmented strategy under an identical execution stack.

Data Collection. We teleoperate the robot with a SpaceMouse to collect expert demonstrations for Push-Wiper. All



(a) Shorter trajectories without large orientation changes

(b) Longer trajectories with large orientation changes

Fig. 3: **PO training trajectories (global sweep plans).** For each stain map, PO plans a single continuous *global* pushing trajectory per iteration to traverse the current stain region, which often yields long paths and large orientation changes. The sponge pose is shown as a blue rectangle at the start and a green rectangle at the end.

demonstrations are performed on planar surfaces, including 150 cleaning tasks: 75 with ketchup and 75 with peanut butter, each comprising 25 simple and 50 complex stain patterns. Each task contains multiple segmented pushes, yielding 448 pushing trajectories in total. Since the task is formulated as segmented pushing on a binary map, each trajectory is represented by three DoFs, $a = (x_b, y_b, \Delta\theta)$. This formulation enables straightforward data augmentation by applying affine transformations consistently to both stain maps and trajectories, producing 2,688 trajectories for policy training.

Implementation. Push-Wiper uses a convolution-based diffusion policy with a DDIM [31] scheduler (sample prediction), trained with 100 diffusion steps and run with 10 steps at inference; we predict n=16 actions per plan. A hybrid force–position controller executes at 100 Hz with F_z^{des}=20 N, and planning terminates once the detected stain area drops below 100 pixels. For fair comparison, all baselines share the same force controller; ASPI is disabled for planar tasks (enabled only for curved-surface generalization); the PO baseline uses identical diffusion settings; and all methods perform one sponge scraping after each executed trajectory.

C. Cleaning results of three methods

For each method, we conduct 20 experiments on ketchup and another 20 on peanut butter, each on distinct stain distributions, with 10 simple and 10 complex cases per stain type. As shown in Table I, Push-Wiper consistently achieves the highest CS and shows strong stability, with an overall average of 89.88, significantly outperforming FC (32.64) and PO (44.98). Notably, while FC and PO suffer performance drops on high-viscosity peanut butter compared to ketchup, Push-Wiper maintains similarly high scores across both stain types.

As shown in Fig. 4, visual inspection reveals clear failure

TABLE I: CS of three methods on ketchup and peanut butter (mean±std over trials).

Method	*Ketchup*			*Peanut butter*			*Overall Avg*
	Simple	**Complex**	**Avg**	**Simple**	**Complex**	**Avg**	
Full-Cover (FC)	54.24±20.98	53.77±11.99	53.99±18.13	16.46±31.41	6.11 ± 38.45	11.28±34.58	32.64±34.76
PushAll-Onetime (PO)	55.33±17.88	60.75 ± 6.60	58.04±13.54	38.41±16.13	25.58±12.90	31.99±15.67	44.98±19.44
Push-Wiper (*Ours*)	**90.43** ± 4.79	**94.18** ± 3.16	**92.30** ± 4.73	**85.79** ± 9.69	**89.13** ± 4.88	**87.46** ± 7.66	**89.88** ± 6.80

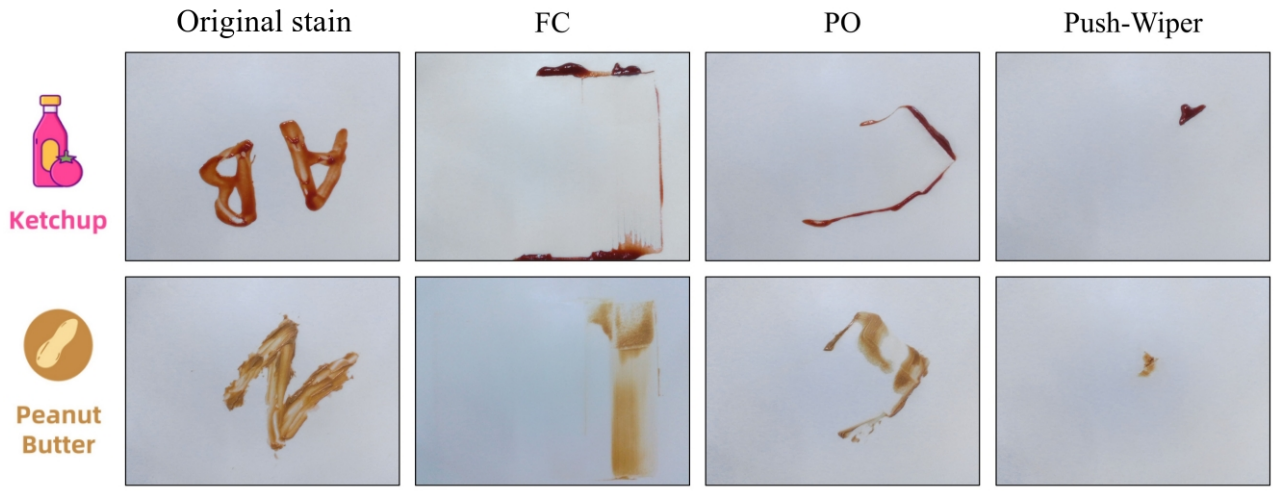


Fig. 4: **Comparison of cleaning results on the same stain distribution using three methods.** Both FC and PO cause varying degrees of secondary contamination, especially on highly viscous peanut butter. In contrast, Push-Wiper consistently achieves superior cleaning performance across all tests.

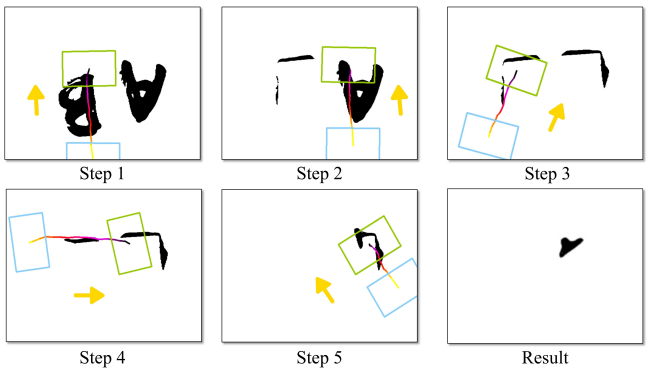


Fig. 5: **Visualization of a segmented pushing process.**

modes of the baselines. **FC** tends to *smear* viscous stains: under force-controlled wiping, sponge deformation drags and redistributes material along the path, causing severe secondary contamination (especially on peanut butter) and leaving substantial residues due to limited absorption, which can even yield negative CS. **PO** adapts its sweep to the current stain distribution, yet it lacks an explicit *aggregation* behavior—its global wipe/push trajectories mainly traverse and spread the stain mass rather than progressively consolidating it—so residual fragments persist across iterations and are difficult to eliminate. In contrast, **Push-Wiper** executes

TABLE II: CS on unseen curved surfaces.

Method	*Convex*			*Concave*		
	Ketch.	**Pb.**	**Avg**	**Ketch.**	**Pb.**	**Avg**
Push-Wiper	95.27 ±0.77	87.63 ±8.31	91.45 ±6.86	94.75 ±2.86	92.10 ±2.58	93.42 ±2.93

step-wise pushes that progressively concentrate stains into compact regions, enabling reliable cleanup even under complex and spatially scattered stain distributions. A segmented pushing process is illustrated in Fig. 5.

Besides cleaning effectiveness, Push-Wiper has an average wall-clock runtime of 130 s per trial, measured from execution start until the stopping criterion (stain area < 100 pixels) is met; this includes the fixed sponge-scraping step after each pushing trajectory. As reported in Table I, we use Push-Wiper's completion time as the matched execution budget T for each test case, and run each baseline for as many full trajectories as fit within T, with sponge scraping after every trajectory. Because secondary contamination can make CS non-monotonic with longer runs (Fig. 4), we report the best CS each baseline attains at any stopping point within T, rather than a time-to-completion metric. Overall, Push-Wiper achieves strong cleaning performance within a minutes-scale execution budget.

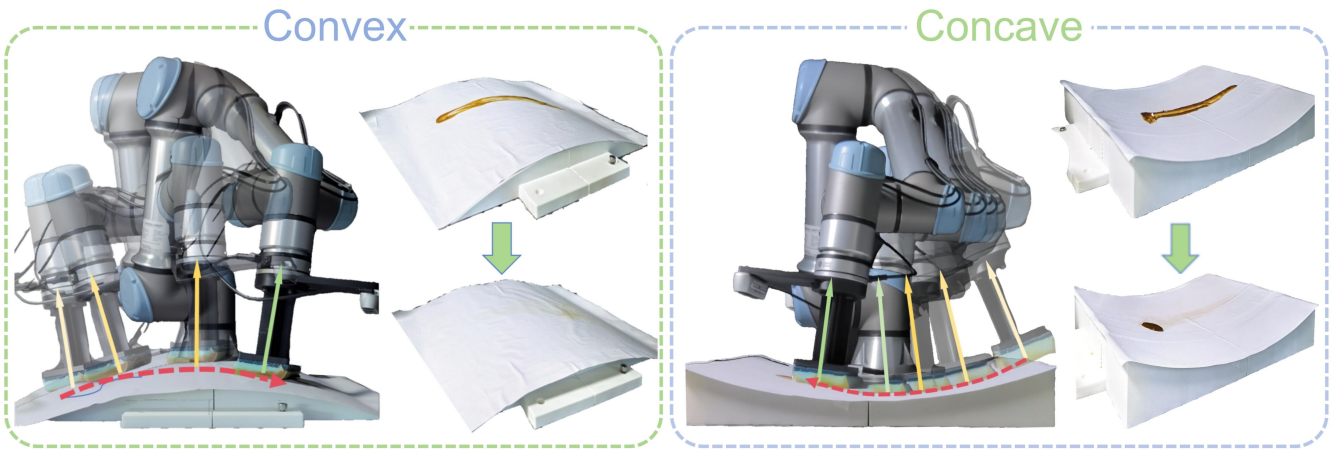


Fig. 6: **Visualization of cleaning tasks on unseen curved surfaces.** On convex and concave surfaces with black pepper sauce and oyster sauce, respectively, both of which are unseen viscous stains.

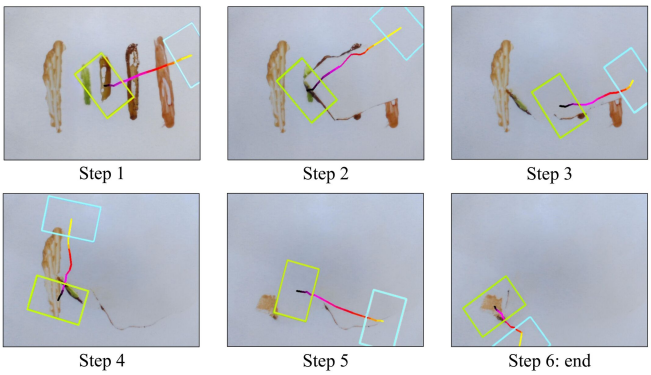


Fig. 7: **A segmented pushing process and result on CVS.** The sponge block is shown as a blue rectangle at the start and a green rectangle at the end.

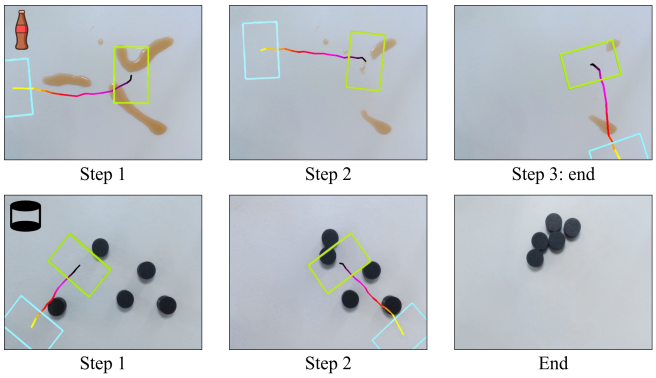


Fig. 8: **Segmented pushing process and results: top row shows cola, bottom row shows black disks.**

TABLE III: CS on unseen objects and stains.

Method	***Solid***	***Liquid***	***CVS***
Push-Wiper	100.00 ± 0.00	92.62 ± 7.43	94.42 ± 3.80

D. Generalization on unseen curved surfaces

We evaluate the generalization of Push-Wiper on curved surfaces using two geometries: a convex and a concave surface. It should be noted that the training data do not include curved surfaces. For each surface, we perform 10 complex-stain trials (5 ketchup, 5 peanut butter). As shown in Table II, Push-Wiper achieves CS of 91.45 and 93.42 on the convex and concave surfaces, respectively, comparable to those on planar surfaces. Fig. 6 shows that Push-Wiper adaptively generates push trajectories, while ASPI keeps the end-effector normal aligned with the surface normal throughout the motion, thereby demonstrating robust generalization to out-of-distribution surface geometries.

E. Generalization on unseen objects and stains

We evaluate Push-Wiper on a variety of unseen scenarios, including solids, liquids, and a combination of unseen viscous stains (abbreviated as CVS), such as black pepper sauce and oyster sauce, to assess its zero-shot generalization to other substances. For the black disks, we randomly place five disks on the table. Since solids cannot be removed from the surface using a sponge, we evaluate whether the policy can aggregate them into a small area. If the final number of connected regions is one, the CS is set to 100; otherwise, it is 0. We conduct 10 complex-type experiments for each of solids, liquids, and CVS. As shown in Table III, Fig. 7 and Fig. 8, Push-Wiper not only generalizes to previously unseen viscous stains but also applies to stains of different physical states.

F. Evaluation of cleaning effectiveness of post-processing

The previous experiments show that the gathering phase already achieves near-complete cleaning. We further employ a post-processing phase to enhance performance. Five trials are conducted for ketchup and peanut butter, respectively. As shown in Table IV, post-processing increases the CS to

TABLE IV: w/o. post-processing vs. w. post-processing

Method	*Ketch.*	*Pb.*	*Avg*
Push-Wiper *(w/o. post-processing)*	93.76	85.12	89.44
Push-Wiper *(w. post-processing)*	$100.00^{\uparrow 6.66\%}$	$98.51^{\uparrow 15.73\%}$	$99.25^{\uparrow 10.97\%}$

100 for ketchup and 98.51 for peanut butter. These results demonstrate that the combination of the gathering phase with the post-processing enables our framework to achieve virtually complete cleaning.

V. Conclusions

In this paper, we present Push-Wiper, a framework for cleaning viscous stains from arbitrary surfaces. Push-Wiper uses a sequence of segmented pushing trajectories to aggregate viscous material and thereby mitigate secondary contamination. A stepwise policy governed by Diffusion Policy adapts to diverse spatial distributions of stains, and the generated actions are executed via our ASPI together with a hybrid force–position controller. In comparative experiments on ketchup and peanut butter, Push-Wiper outperforms baseline methods, achieving an average cleaning score improvement of approximately 130%. We compare results before and after post-processing to verify its effectiveness. Push-Wiper also generalizes to diverse surface geometries and handles solid residues, liquid spills, and unseen viscous stains without retraining. Future work will add collision constraints for safe human–robot interaction and extend post-processing to a broader range of stain types.

References

[1] R. K. Megalingam, S. R. R. Vadivel, S. S. Kotaprolu, B. Nithul, D. V. Kumar, and G. Rudravaram, "Cleaning robots: A review of sensor technologies and intelligent control strategies for cleaning," *Journal of Field Robotics*, 2025.

[2] A. K. Bordoloi, M. F. Islam, J. Zaman, N. Phukan, and N. M. Kakoty, "A floor cleaning robot for domestic environments," in *Proceedings of the 2017 3rd International Conference on Advances in Robotics*, 2017, pp. 1–5.

[3] Z. Li, Q. Xu, and L. M. Tam, "A survey on techniques and applications of window-cleaning robots," *IEEE Access*, vol. 9, pp. 111 518–111 532, 2021.

[4] N. Harmatz, A. Zahra, A. Abdelmalak, S. Purohit, T. Shin, and A. D. Mazzeo, "Hybrid force-position control of an elastic tendon-driven scrubbing robot (tedsr)," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4693–4699.

[5] J. F. Kowalewski, K. Hajjafar, A. Ugent, and J. I. Lipton, "Sccrub: Surface cleaning compliant robot utilizing bristles," *arXiv preprint arXiv:2507.06053*, 2025.

[6] J. R. Landel and D. I. Wilson, "The fluid mechanics of cleaning and decontamination of surfaces," *Annual Review of Fluid Mechanics*, vol. 53, no. 1, pp. 147–171, 2021.

[7] J. 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," *International Journal of Advanced Robotic Systems*, vol. 16, no. 4, p. 1729881419857432, 2019.

[8] S. Wakabayashi, K. Kawaharazuka, K. Okada, and M. Inaba, "Behavioral learning of dish rinsing and scrubbing based on interruptive direct teaching considering assistance rate," *Advanced Robotics*, vol. 38, no. 15, pp. 1052–1065, 2024.

[9] I. Johansson and P. Somasundaran, *Handbook for cleaning/decontamination of surfaces*. Elsevier, 2007.

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

[11] T. Lew, S. Singh, M. Prats, J. Bingham, J. Weisz, B. Holson, X. Zhang, V. Sindhwani, Y. Lu, F. Xia, P. Xu, T. Zhang, J. Tan, and M. Gonzalez, "Robotic table wiping via reinforcement learning and whole-body trajectory optimization," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7184–7190.

[12] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.

[13] J. Hess, G. D. Tipaldi, and W. Burgard, "Null space optimization for effective coverage of 3d surfaces using redundant manipulators," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1923–1928.

[14] Y. Wang and M. Gleicher, "Hierarchically accelerated coverage path planning for redundant manipulators," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 12 098–12 104.

[15] V. Ortenzi, M. Adjigble, J. A. Kuo, R. Stolkin, and M. Mistry, "An experimental study of robot control during environmental contacts based on projected operational space dynamics," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 407–412.

[16] D. Leidner, A. Dietrich, M. Beetz, and A. Albu-Schäffer, "Knowledge-enabled parameterization of whole-body control strategies for compliant service robots," *Autonomous Robots*, vol. 40, no. 3, pp. 519–536, 2016.

[17] N. Hogan, "Impedance control: An approach to manipulation," in *1984 American control conference*. IEEE, 1984, pp. 304–313.

[18] R. Martín-Martín, 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 *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2019, pp. 1010–1017.

[19] C. Tsuji, E. Coronado, P. Osorio, and G. Venture, "Adaptive contact-rich manipulation through few-shot imitation learning with force-torque feedback and pre-trained object representations," *IEEE Robotics and Automation Letters*, 2024.

[20] R. Oishi, S. Sakaino, and T. Tsuji, "Imitation learning based on disentangled representation learning of behavioral characteristics," *arXiv preprint arXiv:2509.04737*, 2025.

[21] M. Song, X. Deng, Z. Zhou, J. Wei, W. Guan, and L. Nie, "A survey on diffusion policy for robotic manipulation: Taxonomy, analysis, and future directions," *Authorea Preprints*, 2025.

[22] B. Zhou, R. Jiao, Y. Li, X. Yuan, F. Fang, and S. Li, "Admittance visuomotor policy learning for general-purpose contact-rich manipulations," *IEEE Transactions on Industrial Electronics*, 2025.

[23] Z. He, H. Fang, J. Chen, H.-S. Fang, and C. Lu, "Foar: Force-aware reactive policy for contact-rich robotic manipulation," *IEEE Robotics and Automation Letters*, 2025.

[24] H. Xue, J. Ren, W. Chen, G. Zhang, Y. Fang, G. Gu, H. Xu, and C. Lu, "Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation," in *Proceedings of Robotics: Science and Systems (RSS)*, 2025.

[25] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[26] Z. Xian, B. Zhu, Z. Xu, H.-Y. Tung, A. Torralba, K. Fragkiadaki, and C. Gan, "Fluidlab: A differentiable environment for benchmarking complex fluid manipulation," in *International Conference on Learning Representations*, 2023.

[27] D. Song and Y. J. Kim, "Distortion-free robotic surface-drawing using conformal mapping," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 627–633.

[28] C. Ott, R. Mukherjee, and Y. Nakamura, "Unified impedance and admittance control," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010, pp. 554–561.

[29] V. Kumbár, S. Ondrušíková, and Š. Nedomová, "Rheological properties of tomato ketchup." 2019.

[30] G. P. Citerne, P. J. Carreau, and M. Moan, "Rheological properties of peanut butter," *Rheologica Acta*, vol. 40, no. 1, pp. 86–96, 2001.

[31] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations (ICLR)*, 2021.