IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Hybrid Contact Dynamics and Residual-RL Framework for Multi-Point Object Pushing

Chen Chen, Xu Dai, and József Kövecses

***Abstract*—Robotic contact manipulation involves applying controlled forces at contact points to guide an object along a desired trajectory while respecting the underlying physical interactions. This letter presents a novel framework that integrates dynamic modeling and Reinforcement Learning (RL) to achieve robust object pushing with a redundant robotic arm. First, a comprehensive dynamic contact model is formulated, incorporating unilateral constraints and a box friction model to capture the nonlinearities present in real-world contact dynamics. Second, the model is extended to handle multiple simultaneous point contacts, enabling effective trajectory planning and tracking for a redundant robotic arm in multi-contact pushing tasks. Third, an RL strategy is introduced as a residual module that augments a model-based controller to improve pushing performance. Simulation and real-world experiments with a Kinova Gen2 arm demonstrate that the proposed method achieves accurate trajectory following and stable contact interactions, significantly outperforming traditional PD control strategies in dynamic pushing scenarios.**

***Index Terms*—Contact modeling, manipulation planning, integrated planning and control, reinforcement learning.**

I. Introduction

PUSHING objects precisely is crucial in robotic manipulation, especially in contact-rich tasks. Robots without grippers, often chosen to save cost or space, rely on pushing as a practical alternative to traditional grasping. Additionally, pushing is essential when manipulating heavy objects, where conventional pick-and-place methods become challenging, even with large grippers. Achieving accurate placement or trajectory tracking through pushing is complicated by unknown object geometries, complex frictional interactions, and the inherent nonlinearity and sensitivity of initial contact conditions [1].

Previous research has extensively studied robotic pushing tasks, often adopting simplified assumptions such as quasi-static conditions or prioritizing path planning without detailed modeling of contact dynamics. For instance, Lynch and Mason [2] analyzed stable quasi-static pushing, identifying directions that maintain continuous object-manipulator contact through heuristic methods. Miyazawa et al. [3] employed the rapidly-exploring random tree (RRT) algorithm [4] to plan flexible contacts for pushing planar objects. Lee et al. [5] developed a hierarchical approach that segments pushing tasks into selecting object contact points, pose estimation, and manipulator contact planning.

Recent work has increasingly utilized high-fidelity datasets for data-driven planar pushing. Researchers like Lynch [2] and Yoshikawa [6] used sensor-vision feedback to estimate dynamic parameters such as friction distribution. Yu et al. [1] assembled extensive push datasets with systematic variations in object shape, materials, and velocities to learn contact behaviors. Building upon these datasets, Zhou et al. [7] proposed a rapid stochastic contact model, while Bauza et al. [8] developed probabilistic predictions of planar push outcomes using Gaussian processes. However, these methods rely on different sensors and do not explicitly address scenarios with complex, multi-contact dynamics.

Learning-based methods have recently advanced planar manipulation but typically involve restrictive assumptions. Bauza and Rodriguez [9] presented a data-efficient push planner with local Gaussian processes. Yang et al. [10] learned tactile-based RL policies transferable to novel objects. Chi et al. [11] used diffusion models conditioned on target end poses. Cong et al. [12] trained a recurrent forward model in simulation under quasi-static conditions, employing MPC for control. Dengler et al. [13] utilized DRL frameworks for goal-directed pushing in cluttered environments based on depth and contact feedback. Jin et al. [14] examined the sensitivity of transformer models to physical and visual domain shifts in trajectory predictions, highlighting their high data dependence. Although promising, these methods: (i) require large-scale data or additional sensing modalities, (ii) optimize mainly for target pose and leave the time-parameterized trajectory implicit, an issue for contact-rich pushing where this can break continuous contact, induce abrupt accelerations, and yield inconsistent force profiles, and (iii) neglect comprehensive multi-contact friction dynamics.

To overcome the limitations of purely model-based or model-free methods in planar pushing tasks, we propose a Hybrid Contact Dynamics and Residual-RL Framework. The term Hybrid denotes the integration of analytical contact modeling with data-driven residual learning. Our contributions are as follows:

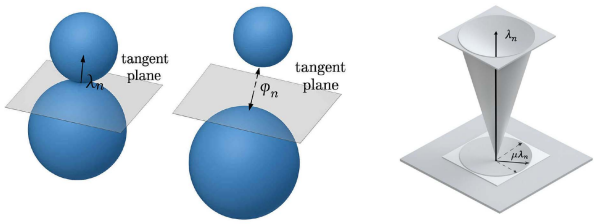
Received 11 July 2025; accepted 4 November 2025. Date of publication 19 November 2025; date of current version 27 November 2025. This article was recommended for publication by Associate Editor A. Gams and Editor J. Borràs Sol upon evaluation of the reviewers' comments. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, in part by CM Labs Simulations, Inc., in part by Canadian Space Agency, and in part by the National Research Council of Canada. The support is gratefully acknowledged. *(Corresponding author: Chen Chen.)*

Chen Chen and Xu Dai are with the Department of Mechanical Engineering and the Center for Intelligent Machines, McGill University, Montreal, QC H3A 0C3, Canada (e-mail: chen.chen12@mail.mcgill.ca; xu.dai@mail.mcgill.ca).

József Kövecses is with the Department of Mechanical Engineering and the Center for Intelligent Machines, McGill University, Montreal, QC H3A 0C3, Canada, and also with the University Research and Innovation Center (EKIK), Obuda University, 1034 Budapest, Hungary (e-mail: jozsef.kovecses@mcgill.ca).

Digital Object Identifier 10.1109/LRA.2025.3634905

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



(a) Unilateral constraint. (b) Box friction model.

Fig. 1. Constraint modeling: (a) Unilateral constraint and (b) box friction model.

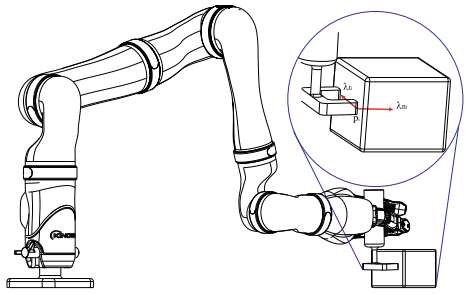


Fig. 2. Schematic of the planar multi-contact pushing configuration. Multiple contact points p_i apply planar forces $\{\lambda_{n_i}, \lambda_{t_i}\}$ to the object. All contacts are positioned at the same z-height.

TABLE I
COMPARISON WITH PRIOR ROBOTIC PUSHING METHODS

Aspect	Previous Methods	Our Method
Dynamics Modeling	Quasi-static assumptions; simplified or linear friction models	Full dynamic modeling with unilateral constraints and box friction
Contact Handling	Single-point or multi-face contacts	Multiple contacts on the same face with force distribution
Push Target	Typically final pose and limited trajectory consideration	Precise trajectory tracking
RL Reward Design	Sparse reward signals and sensor-based rewards	Force-level rewards derived from contact dynamics; no external sensing required

1) Introduce a comprehensive dynamic contact model for a redundant robotic arm that includes unilateral constraints [15] and a box friction model (see Fig. 1) [16], [17]. This approach handles contact dynamics that are typically nonlinear and challenging in real-world tasks.
2) Extend the model to accommodate multiple point contacts, enabling a redundant robotic arm to push an object along a specified trajectory.
3) Integrate RL not as a standalone model-free controller, but as a residual learning module that fine-tunes control inputs on top of the model-based controller. We design a novel force-based reward function that enables the RL agent to adaptively correct errors due to friction uncertainties and contact disturbances.

Table I summarizes key differences compared to existing methods.

II. METHODS

A. Contact Dynamics Modeling

Pushing an object along a predefined trajectory and achieving accurate positioning is a contact-rich manipulation problem, as illustrated in Fig. 2. The dynamics of any robotic arm under such conditions can be described by the following equations:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{c} = \mathbf{f}_a + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda} \tag{1}$$

$$\mathbf{A}\dot{\mathbf{q}} = \mathbf{w} \tag{2}$$

where $\mathbf{M}$ is the joint space mass matrix, and $\mathbf{c}$ includes centrifugal and Coriolis forces. $\dot{\mathbf{q}}$ denotes the joint velocities, and $\mathbf{f}_a$ represents the control input torques. The matrix $\mathbf{A}$ is the constraint Jacobian, which maps joint velocities to constraint-space velocities $\mathbf{w}$. The contact forces $\boldsymbol{\lambda}$ stack the normal and tangential components applied at multiple contact points:

$$\boldsymbol{\lambda} = [\lambda_{n_1}, \lambda_{t_1}, \ldots, \lambda_{n_k}, \lambda_{t_k}]^{\mathrm{T}}, \quad i = 1, \ldots, k \tag{3}$$

where λ_{n_i} and λ_{t_i} represent the normal and tangential forces at the i^{th} contact point. Given the planar nature of the pushing task, tangential forces λ_{t_i} significantly influence planar translation and rotation about the vertical (z) axis; the orthogonal tangential direction is omitted. Similarly, the relative constraint velocities are:

$$\mathbf{w} = [w_{n_1}, w_{t_1}, \ldots, w_{n_k}, w_{t_k}]^{\mathrm{T}}, \quad i = 1, \ldots, k \tag{4}$$

where w_{n_i} and w_{t_i} denote the normal and tangential constraint velocities at contact point i. To model contact interactions, we adopt the unilateral constraint and the box friction model. The unilateral constraint enforces non-penetration between the pusher and the object, expressed via a linear complementarity relationship between the normal relative velocity and the corresponding normal contact force. The box friction model, derived from the Coulomb friction cone [18], discretizes the friction constraint within a bounded box along two orthogonal tangential directions. The friction bounds are defined using a coefficient vector $\boldsymbol{\mu} = [\mu_1 \ldots \mu_k]^{\mathrm{T}}$, and the resulting friction forces satisfy:

$$\lambda_{t_i} \in [-\mu_i \lambda_{n_i}, +\mu_i \lambda_{n_i}], \quad i = 1, \ldots, k \tag{5}$$

where μ_i is the friction coefficient at the i^{th} contact point. While the box friction model is an approximation of the true quadratic Coulomb friction cone, we choose it for its trade-off between physical fidelity and computational efficiency. By representing friction limits as linear inequalities, the contact dynamics can be formulated as a Mixed Linear Complementarity Problem (described subsequently) rather than a nonlinear one, making it more tractable for real-time control and ensuring a unique solution at each time step [16], [17]. To simulate the contact interaction, we discretize the dynamics at the velocity level using a first-order method. Over a time step h from t to t^+, the velocity

update is approximated as:

$$\ddot{\mathbf{q}} \approx \frac{\dot{\mathbf{q}}^{+} - \dot{\mathbf{q}}}{h} \tag{6}$$

Introducing time discretizations, unilateral constraint, and box friction model, the system is formed as:

$$\begin{bmatrix} \mathbf{M} & -\mathbf{A}^{\mathrm{T}} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}^{+} \\ h\boldsymbol{\lambda}^{+} \end{bmatrix} + \begin{bmatrix} -\mathbf{M}\dot{\mathbf{q}} + h\mathbf{c} - h\mathbf{f}_a \\ h\dot{\mathbf{A}}\dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{w}^{+} \end{bmatrix} \tag{7}$$

with the complementarity condition:

$$\begin{aligned} \mathbf{0} \le \mathbf{w}_n^{+} \perp \boldsymbol{\lambda}_n^{+} \ge \mathbf{0} \\ \mathbf{w}_t^{+} \perp \boldsymbol{\lambda}_t^{+} \in [-\boldsymbol{\mu}\boldsymbol{\lambda}_n^{+}, +\boldsymbol{\mu}\boldsymbol{\lambda}_n^{+}] \end{aligned} \tag{8}$$

The contact dynamic model, formulated as a Mixed Linear Complementarity Problem (MLCP), serves as the computational backbone of our system. Unlike prior work that often relies on simplified or quasi-static approximations, our formulation captures the underlying physics of multi-point frictional contact with high fidelity. It allows joint velocities and contact forces to be computed together within a single formulation, ensuring physical consistency and numerical stability. This unified approach makes the model directly usable in both simulation and control pipelines. To the best of our knowledge, this is the first use of such a dynamic model in an object pushing scenario using a redundant robotic arm. As such, it provides a reliable foundation for learning and executing complex contact-rich tasks in both simulation and real-world settings.

B. Trajectory Planning and PD Control

Accurate trajectory planning and appropriate control selection are crucial for achieving reliable performance in contact tasks. A general approach can be structured into three main stages: (i) objectlevel trajectory generation, (ii) mapping of the trajectory into desired contact force distributions and contact positions, and (iii) robotic joint space control.

1) Object Trajectory Planning: To effectively plan the object's motion during the pushing task, we employ the instant center of rotation (ICR) method [19]. The ICR is a conceptual point in the plane of a moving rigid body at which the instantaneous velocity is zero. Under this formulation, the motion of every other point on the object can be interpreted as pure rotation around the ICR. By leveraging the ICR representation, the combined translational and rotational motion of the object is transformed into a single continuous rotation about a moving center. This approach enables smooth, coordinated trajectory generation from an initial pose to a target pose without requiring piecewise interpolation or switching between different motion primitives. Moreover, it provides a natural way to design feasible pushing trajectories that maintain stable contact, minimize abrupt velocity changes, and reduce control complexity.

2) Force Distributions: Once the desired object trajectory is defined, the primary challenge is determining the magnitudes and positions of contact forces that the robotic arm should apply to achieve this motion. At the i^{th} contact point, the desired applied force is expressed as:

$$\mathbf{F}_{i,\text{des}} = F_{n_i,\text{des}}\mathbf{e}_n + F_{t_i,\text{des}}\mathbf{e}_t \tag{9}$$

where scalar magnitudes $F_{n_i,\text{des}}$ and $F_{t_i,\text{des}}$ represent the normal and tangential force components, and unit vectors $\mathbf{e}_n$ and $\mathbf{e}_t$ indicate local normal and tangential directions respectively. Under Coulomb friction, the following friction cone must hold:

$$F_{n_i,\text{des}} \ge 0, \quad |F_{t_i,\text{des}}| \le \mu_i F_{n_i,\text{des}}, \quad i = 1, \ldots, k \tag{10}$$

The object's dynamics are governed by Newton–Euler equations, relating applied forces to translational and rotational accelerations, all expressed consistently in the local coordinate frame:

$$\mathbf{M}_{\text{ob}}\dot{\mathbf{v}}_{\text{ob,des}} + \mathbf{c}_{\text{ob}} = \mathbf{A}_{\text{ob}}^{\top}\mathbf{F}_{\text{des}} + \mathbf{f} \tag{11}$$

$$\mathbf{A}_{\text{ob}}\mathbf{v}_{\text{ob,des}} = \mathbf{w}_{\text{p}} \tag{12}$$

where $\mathbf{M}_{\text{ob}} = \text{diag}(m, m, I_z)$ is the mass matrix, $\dot{\mathbf{v}}_{\text{ob,des}} = [\dot{v}_{n,\text{ob,des}}, \dot{v}_{t,\text{ob,des}}, \dot{\omega}_{z,\text{ob,des}}]^{\top}$ represents desired generalized accelerations at the object's center of mass, and $\mathbf{c}_{\text{ob}}$ captures Coriolis and centrifugal terms. The Jacobian matrix $\mathbf{A}_{\text{ob}}$ transforms generalized velocities at the object's center of mass into translational velocities $\mathbf{w}_{\text{p}}$ at all k contact points. The vector $\mathbf{F}_{\text{des}} = [F_{n_1,\text{des}}, F_{t_1,\text{des}}, \ldots, F_{n_k,\text{des}}, F_{t_k,\text{des}}]^{\top}$ represents the set of desired normal and tangential forces for all k contact points. The vector $\mathbf{f} = [f_n, f_t, \tau_z]^{\top}$ denotes external friction forces and torque resulting from ground interaction. Each contact point's position relative to the object's center of mass, in the local frame, is given by $\mathbf{r}_i = n_i\mathbf{e}_n + t_i\mathbf{e}_t$. To compute optimal forces and contact positions, we solve the constrained optimization problem:

$$\begin{aligned} \min_{\{F_{n_i,\text{des}}, F_{t_i,\text{des}}, n_i, t_i\}} \quad & \left\| \mathbf{M}_{\text{ob}}\dot{\mathbf{v}}_{\text{ob,des}} + \mathbf{c}_{\text{ob}} - \mathbf{A}_{\text{ob}}^{\top}\mathbf{F}_{\text{des}} - \mathbf{f} \right\|^2 \\ \text{s.t.} \quad & F_{n_i,\text{des}} \ge 0, \quad |F_{t_i,\text{des}}| \le \mu_i F_{n_i,\text{des}}, \\ & (n_i, t_i) \in \mathcal{R}, \quad i = 1, \ldots, k \end{aligned} \tag{13}$$

where $\mathcal{R}$ denotes the admissible set of contact locations on the object's surface, ensuring that all contact points remain within the object's geometry and are physically realizable given the pusher's configuration. In practice, $\mathcal{R}$ is computed as the intersection between feasible object-surface contact regions and the spatial constraints imposed by the pusher's geometry. The resulting optimized parameters serve as direct inputs for robotic trajectory planning and control.

It is worth noting that the cube in Fig. 3 serves as a simplified 3-D illustration. The proposed optimization framework for force distribution and contact selection is applicable to a wide range of convex rigid bodies, such as cuboids, polyhedra, and cylinders, where surface normals and contact geometries are well defined. While the approach extends to arbitrary convex shapes, nonconvex or geometrically complex objects may require additional constraints or tailored numerical methods to ensure feasible and accurate solutions.

3) Operational Space Velocity Planning and PD Controller: Given the desired object trajectory and the computed contact

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

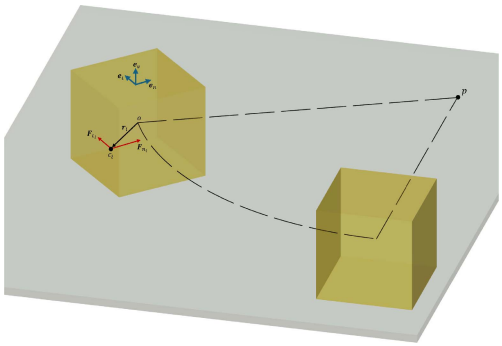


Fig. 3. Illustration of pushing an object at a representative contact point c_i. $F_{n_i,\text{des}}$ and $F_{t_i,\text{des}}$ are the desired normal and tangential force represented at the local frame. The contact point c_i is located on one face of the object at the same vertical height as its center of mass o. p denotes the instant center of rotation (ICR) that constrains the object's motion.

force distribution, the final stage of the control pipeline involves commanding the robot's end-effector motions to realize the planned object behavior. At each control cycle, the desired end-effector velocity $\mathbf{v}_{e_{\text{des}}}$ is computed to guide the end-effector toward the target contact point. This operational space velocity must be translated into joint level velocity commands that the robot can execute. For redundant manipulators, inverse kinematics [20] computes joint velocities to achieve the desired end-effector motion:

$$\dot{\mathbf{q}}_{\text{des}} = \mathbf{J}^{\dagger}\mathbf{v}_{e_{\text{des}}} \tag{14}$$

where $\mathbf{J}^{\dagger}$ is the Moore–Penrose pseudoinverse of the Jacobian matrix $\mathbf{J}$. The null space velocity command is not utilized in this push task scenario. Joint space tracking of the computed reference trajectories is enforced through a torque-level proportional-derivative (PD) control scheme:

$$\boldsymbol{\tau} = \mathbf{K}_p(\mathbf{q}_{\text{des}} - \mathbf{q}) + \mathbf{K}_v(\dot{\mathbf{q}}_{\text{des}} - \dot{\mathbf{q}}) \tag{15}$$

where $\mathbf{q}$ and $\dot{\mathbf{q}}$ are the current joint positions and velocities, and $\mathbf{K}_p, \mathbf{K}_v$ are positive definite control gains tuned to ensure convergence and stable tracking. The command torque $\boldsymbol{\tau}$ is applied as external forces to the robot's dynamic model (as described in (7)), resulting in joint velocities and subsequent motion of the end-effector. Through this closed-loop structure, deviations in joint space positions and velocities are continually corrected by torque adjustments.

4) Limitations of PD Controller in Force Regulation: Despite this structured control pipeline, the actual contact forces exerted by the robot still differ from the desired ones computed by the contact dynamic model in (7). Contact forces are not directly commanded but instead arise from the nonlinear interaction dynamics between the robot and the object. If, at time t, the realized contact forces deviate from the desired values, the object may not reach the expected configuration at time t^+. This deviation can lead to drift in the contact location and loss of alignment with the precomputed trajectory. In other words, relying solely on a PD motion controller is insufficient to guarantee precise contact force application [21] or accurate trajectory tracking in dynamic pushing tasks. This limitation of purely position-based control is well recognized in robotics research, prompting the development of other force control approaches [22], [23].

Algorithm 1: Residual-RL Assisted Multi-Contact Pushing.

Require: Nominal policy π_H (hand-engineered, tracks ICR-based planned trajectory and desired force distribution $\mathbf{F}_{\text{des}}$); residual policy π_θ (trained RL policy for corrections)

Initialize: observe current robot state $(\mathbf{q}, \dot{\mathbf{q}})$, nominal contact set $\mathbf{r}$, object position $\mathbf{p}$, and contact forces $\boldsymbol{\lambda}$
for $n = 0$ **to** $N - 1$ Steps **do**
  $\mathbf{v}_e^H \leftarrow \pi_H(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{r})$ // nominal end-effector velocity
  if object pose deviates from ICR trajectory **then**
    $\mathbf{v}_e^\theta \leftarrow \pi_\theta(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{r}, \boldsymbol{\lambda}, \mathbf{F}_{\text{des}})$ // residual velocity
  else
    $\mathbf{v}_e^\theta \leftarrow \mathbf{0}$ // no residual correction needed
  end if
  $\mathbf{v}_e \leftarrow \mathbf{v}_e^H + \mathbf{v}_e^\theta$ // combined end-effector velocity command
  $\dot{\mathbf{q}}_{\text{des}} \leftarrow \mathbf{J}^\dagger \mathbf{v}_e$; $\mathbf{q}_{\text{des}} \leftarrow \mathbf{q} + \dot{\mathbf{q}}_{\text{des}}\,\Delta t$ // inverse kinematics
  $\boldsymbol{\tau} \leftarrow \mathbf{K}_p(\mathbf{q}_{\text{des}} - \mathbf{q}) + \mathbf{K}_v(\dot{\mathbf{q}}_{\text{des}} - \dot{\mathbf{q}})$ // PD controller
  $(\mathbf{q}^+, \dot{\mathbf{q}}^+, \mathbf{p}^+, \boldsymbol{\lambda}^+) \leftarrow \text{MLCP}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{p}, \boldsymbol{\tau})$ // simulate one time-step with our contact dynamic model
  $(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{p}, \boldsymbol{\lambda}) \leftarrow (\mathbf{q}^+, \dot{\mathbf{q}}^+, \mathbf{p}^+, \boldsymbol{\lambda}^+)$ // update state for next step
end for

C. RL Assistance

1) Residual Velocity Learning: To overcome the limitations of this velocity- and position-based PD controllers in achieving precise contact forces and contact locations, we introduce a RL policy that provides augmentation of the end-effector velocity. The central concept is to learn a residual velocity correction term, enabling the robot to dynamically compensate for deviations from the planned motion and desired force profiles. Formally, the commanded end-effector velocity is defined as:

$$\mathbf{v}_e = \mathbf{v}_e^H + \mathbf{v}_e^\theta \tag{16}$$

where $\mathbf{v}_e^H$ is the nominal velocity computed by the hand-engineered PD controller, and $\mathbf{v}_e^\theta$ is the RL-generated residual velocity. This residual term addresses real-time discrepancies in the robot's motion, particularly slight misalignments in contact positions or inaccuracies in the contact forces.

Critically, velocity corrections introduced by the residual policy not only affect the end-effector's position but also directly influence the contact forces applied to the object. Specifically, adjustments in the end-effector velocity translate into corresponding changes in joint velocities. These joint velocity variations, in turn, produce adjustments in joint torques through the PD control law (15). Because these torques serve as external control inputs within the contact dynamics model (7), any alteration in joint torques immediately impacts the contact forces. Consequently, improper residual velocity adjustments can lead to unintended contact forces, causing the object to deviate from

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE II
PHYSICAL PROPERTIES OF THE PUSHER AND OBJECTS

	Cube	Wedge
Pusher Mass	30 g	40 g
Pusher Dimensions	[104 × 48 × 30] mm	
Object Mass	300 g	350 g
Object Dimensions	[88 × 88 × 88] mm	[140 × 85 × 80] mm

its desired trajectory. In extreme scenarios, such adjustments may even result in the loss of stable contact.

2) Reward Function Design: The reward function is designed to align with the physical objectives of the pushing task, emphasizing accurate force application over pure trajectory tracking. Specifically, it penalizes deviations between the actual and desired contact forces and torques across all k contact points at n timestep:

$$R = -\sum_{j=1}^{n}\sum_{i=1}^{k}\Big[w_1|F^j_{n_{i,\text{des}}} - \lambda^j_{n_i}| + w_2|F^j_{t_{i,\text{des}}} - \lambda^j_{t_i}| + w_3|\tau^j_{z_{i,\text{des}}} - \tau^j_{z_i}|\Big]. \quad (17)$$

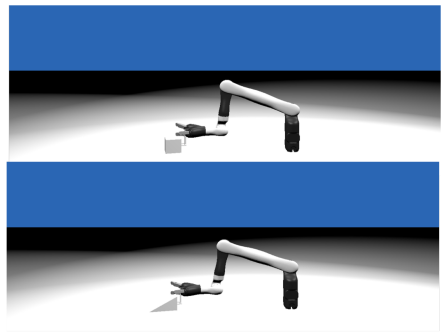
where $\lambda^j_{n_i}$, $\lambda^j_{t_i}$, and $\tau^j_{z_i}$ denote the actual contact normal force, tangential force, and torque at the i^{th} contact point at timestep j, respectively. The terms with subscript "des" indicate the corresponding desired values computed from our dynamic optimization. The scalar weights w_1, w_2, and w_3 are introduced to balance the relative importance between translational and rotational tracking objectives.

This learning strategy is inspired by Residual Policy Learning (RPL) [24], which combines model-based controllers with data-driven corrections. However, in contrast to prior applications that focus on trajectory-based or sparse rewards, we integrate Newton–Euler dynamics into the reward formulation to guide learning at the force level. Moreover, the reward computation relies solely on internal state variables available in simulation, avoiding the need for external sensing such as vision and force sensors. The entire training process is conducted in a simulated environment using our dynamic model, and the learned residual velocities are directly deployed on the real robot without additional adaptation.

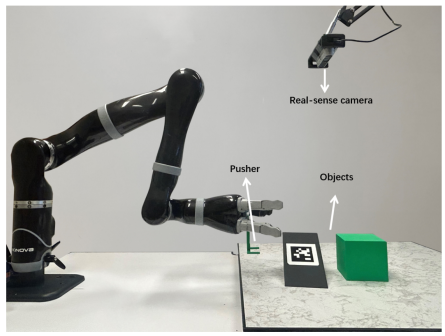
III. EXPERIMENTAL SETUP

We evaluate our proposed method through experiments using a Kinova Gen2 7-DOF robotic arm to perform planar pushing with two different objects (a cube and a wedge shape). A custom two-point pusher is mounted on the robot's end-effector to interact with the objects. The simulation and real-world setups are shown in Fig. 4, and the key physical parameters of the objects and pusher are summarized in Table II.

The primary objective of our experiments is to validate the robot's ability to accurately track entire planned trajectories generated by the ICR method across a variety of target configurations. In each trial, we randomize the object's target pose, including its final (x, y) position and orientation, within



(a) Simulation environment in Vortex Studio.



(b) Real experiment setup.

Fig. 4. Pushing task scenarios in simulation and real world.

predefined ranges. Given any specific start and end pose, the ICR planner produces a unique circular arc trajectory; thus, this randomization ensures each trial evaluates a distinct trajectory. To increase motion complexity, we evaluate three angular velocity profiles along the ICR trajectory: constant (uniform rotation), trapezoidal (acceleration, steady speed, and deceleration) [25], and chirp-like (oscillating rotation rate that increases over time) [26]. These profiles share the same geometric arc but vary the temporal characteristics of rotation, introducing different dynamic challenges for trajectory tracking.

A. Simulation Environment

The simulation experiments are conducted using Vortex Studio, which accurately models complex contact dynamics consistent with our derived formulation (7). Initially, the pusher is moved from the home position to the initial contact points using an inverse kinematics solution. These pre-contact motions rely purely on model-based planning to achieve the nominal contact configuration. Once the pushing phase begins, the system continuously monitors the interaction forces. If the object pose deviates from the ICR trajectory, the RL-based residual velocity correction is activated to augment the end-effector velocity.

During simulation training, we expose the RL agent to both object types and to the full range of trajectory profiles to ensure broad generalization. Each training episode randomly selects an object shape and a target pose, resulting in different ICR trajectories with one of the three angular velocity profiles. This diversity of training scenarios allows the learned residual policy to handle different objects and trajectories. Furthermore, to

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

bridge the gap between simulation and reality, domain randomization techniques [27] [28] are systematically applied to enhance robustness against uncertainties and model discrepancies in simulation environment. Specifically, we randomize physical properties such as object mass, pusher mass, contact friction, and ground friction coefficients within realistic bounds. We employ the Deep Deterministic Policy Gradient (DDPG) algorithm [29], [30] to learn this residual velocity policy. The actor network receives a compact state input comprising actual and desired contact forces, end-effector velocities, and object positions, outputting the continuous end-effector velocity correction $\mathbf{v}_e^\theta$. The critic network evaluates the quality of the selected action using a temporal-difference error and guides the policy update. Both actor and critic networks consist of fully connected layers with ReLU activations and are trained using an off-policy replay buffer with soft target updates. To encourage exploration, Ornstein–Uhlenbeck (OU) noise [31] is added to the actor output during training. The raw actions are normalized to $[-1, 1]$ and then scaled by an action coefficient to produce feasible velocity corrections at the end-effector. The training is executed within this environment on a workstation equipped with an NVIDIA RTX 3060 GPU.

B. Real-World Experimental Platform

In real-world validation, we utilize the same PLA-based pusher and objects from the simulation setup to maintain consistency in geometry and inertial properties. The Kinova Gen2 robotic arm executes commands at a high-frequency control loop of 100 Hz. Pose data of objects are obtained using an Intel RealSense D455 RGB-D camera mounted above the experimental workspace. To achieve reliable pose estimation, ArUco markers are affixed to the top face of the objects to reduce measurement noise [32]. The camera provides pose measurements at 30 Hz, necessitating synchronization strategies to align lower-frequency camera data with the high-frequency robotic control loop. Specifically, object poses are recorded every three control steps to match the camera's frame rate, ensuring coherent comparisons between simulated and real-world trajectories. Notably, the camera system is employed solely for post-experiment performance evaluation; it does not serve as feedback for the robot's control policy during real-time execution.

IV. Results

In this section, we evaluate our proposed RL assistance approach by comparing its performance against the PD controller. We focus on both simulation and real-world experimental results, analyzing trajectory-tracking accuracy and contact force performance. In addition, we assess the accuracy of the contact dynamic model by comparing simulated predictions with real-world measurements.

A. Simulation Training

Each training episode runs for 20 s, equivalent to 2000 simulation time steps. For each episode, we evaluate the performance of PD-only baseline and RL assistance approach by calculating

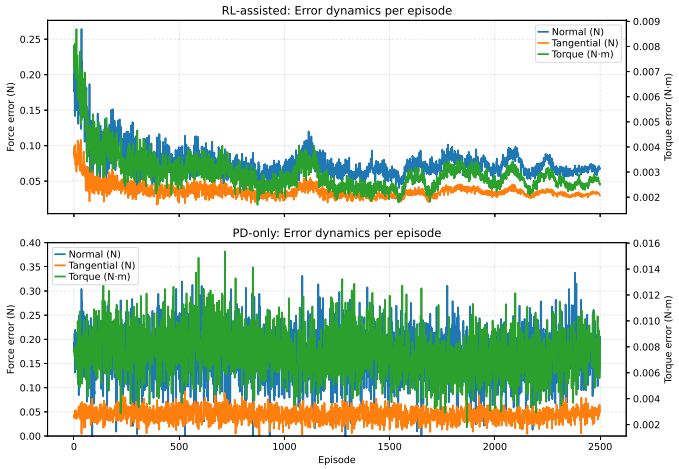
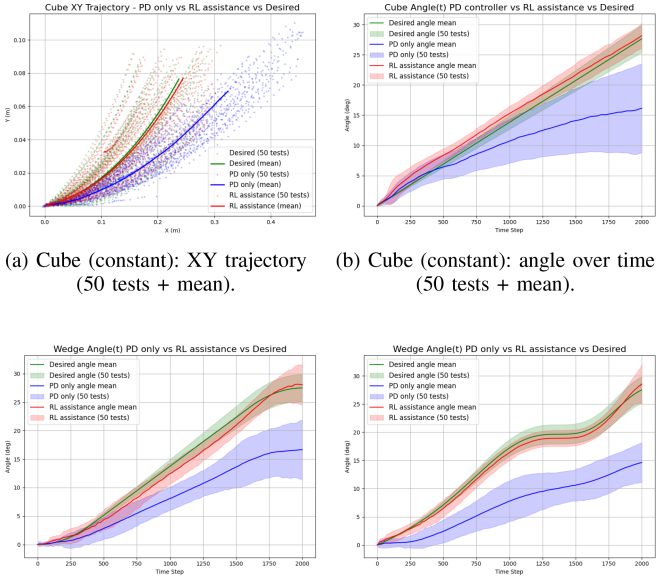


Fig. 5. Error dynamics during the training. Each point is the per-episode mean absolute error (averaged over the 2000 time steps in that episode). Left axis: force errors (normal, tangential); right axis: torque error.



(a) Cube (constant): XY trajectory (50 tests + mean).

(b) Cube (constant): angle over time (50 tests + mean).

(c) Wedge (trapezoid): angle over time (50 tests + mean).

(d) Wedge (chirp): angle over time (50 tests + mean).

Fig. 6. Trajectory/angle tracking comparison. Green: desired; blue: PD-only; red: RL-assisted. Means are computed over 50 trials per condition.

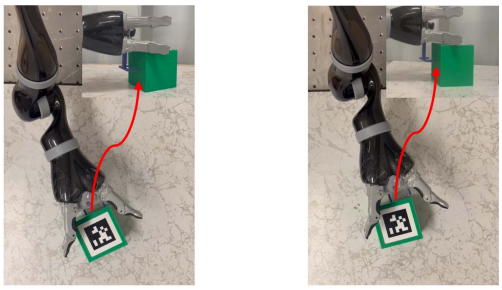
the mean absolute tracking errors in normal force, tangential force, and torque. Fig. 5 presents these per-episode errors across 2500 episodes. The RL-assisted approach demonstrates a clear reduction in all error metrics within the initial hundred episodes, achieving stable convergence around episode 1500. In contrast, the PD-only controller maintains higher errors and greater noises, indicating its limited ability to adapt to different object geometry and angular velocity profiles.

B. Trajectory Tracking Accuracy (Real Experiments)

Fig. 6 summarizes experimental tracking performance comparing our RL assistance approach with a PD-only baseline over 50 real-world pushing trials per scenario. The designed

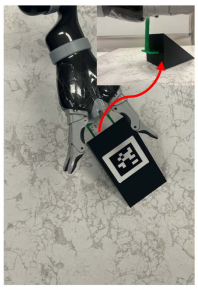
TABLE III
MEAN TRACKING PERCENTAGE ERRORS FOR EACH OBJECT AND ANGULAR TRAJECTORY OVER 50 REAL-WORLD TRIALS

Object	Angular Trajectory	Translation Error (%)		Angle Error (%)	
		PD-only	RL-assisted	PD-only	RL-assisted
Cube	Constant	22.3	4.0	14.5	4.0
	Trapezoid	25.8	4.2	16.7	4.8
	Chirp	26.5	4.8	17.4	5.2
Wedge	Constant	23.0	4.2	13.3	5.5
	Trapezoid	27.9	4.5	15.8	4.8
	Chirp	29.6	4.8	18.7	4.3

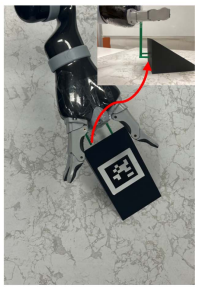


(a) Cube—constant ICR: full contact.

(b) Cube—constant ICR: contact loss.



(c) Wedge—trapezoidal ICR: full contact.



(d) Wedge—trapezoidal ICR: contact loss.

Fig. 7. Representative contact conditions under RL-assisted control (left) and PD-only baseline (right) for cube (top) and wedge (bottom) objects.

objects are tested along different ICR trajectories. For clarity, representative trajectories are selected to illustrate key variations: (i) translational and rotational tracking of the cube under a constant angular velocity (Fig. 6(a) and (b)), and (ii) rotational tracking of the wedge under trapezoidal and chirp angular velocity profiles (Fig. 6(c) and (d)). In all selected examples, the RL-assisted controller achieves notably closer adherence to the desired trajectory compared to the PD baseline. Table III summarizes mean percentage tracking errors, computed over the entire trajectory and averaged over 50 trials per scenario, clearly showing RL assistance approach improvements across all shapes and profiles.

Fig. 7 presents two representative real-world experimental trials for each object under different ICR rotation trajectories, clearly demonstrating the contrast between stable and unstable contact conditions. Under conditions where the PD controller produces inaccurate or suboptimal contact forces and positions, the pusher may lose contact at one of its points. This loss reduces

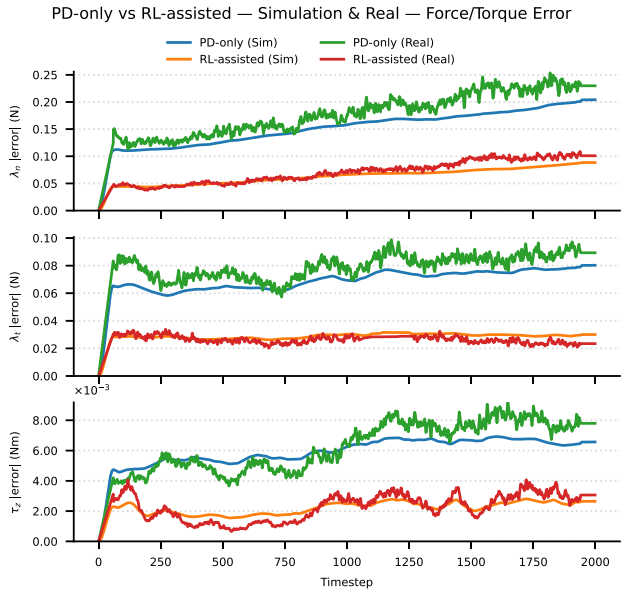


Fig. 8. Cube object, constant angular velocity ICR trajectory: sim–to–real comparison of planar force and torque tracking errors at each time step (mean over 50 trials).

TABLE IV
MEAN PLANAR FORCE/TORQUE TRACKING ERRORS

Object	Angular Trajectory	Force Error (10^{-1} N)				Torque Error (10^{-3} Nm)			
		PD-only		RL-assisted		PD-only		RL-assisted	
		Sim	Real	Sim	Real	Sim	Real	Sim	Real
Cube	Constant	1.71	1.85	0.65	0.73	6.02	6.34	2.26	2.52
	Trapezoid	1.88	2.02	0.72	0.81	6.67	7.21	2.62	2.96
	Chirp	2.05	2.18	0.78	0.89	7.44	7.81	2.93	3.52
Wedge	Constant	1.95	2.08	0.74	0.83	6.51	6.91	2.47	3.13
	Trapezoid	2.12	2.56	0.81	0.92	7.14	7.61	2.18	2.63
	Chirp	2.28	2.43	0.88	0.94	7.95	8.33	2.08	2.48

both the applied force and torque on the object, leading directly to deviations from the intended ICR trajectory. Consequently, the object's rotational motion slows noticeably relative to the desired trajectory.

C. Force and Torque Error (Sim–to–Real Transfer)

To further validate our residual policy approach and the contact dynamic model, we evaluate planar contact forces and torque in simulation and real-world experiments. Because no external force–torque sensor is available in the experimental setup, we utilize the Kinova arm's internal force/torque estimates only for offline validation after real-world trials. Crucially, these internal readings are never used as inputs to the residual policy during deployment. The policy operates entirely without force sensing on the physical robot, relying on force feedback only in simulation training as part of the reward function. Fig. 8 and Table IV summarize the force and torque tracking evaluation. Fig. 8 presents a representative scenario (cube with constant angular rotation), displaying the average force and torque tracking errors at each time step across 50 trials, separately for simulation and real-world experiments under PD controller and

RL assistance approach. In contrast, Table IV provides aggregated scalar metrics, each representing the mean tracking error computed by averaging absolute force and torque deviations over all 2000 time steps, and then further averaging these results across the 50 trials. Two key conclusions can be drawn from this analysis. First, under the PD-only controller, the similarity between simulation and real-world results indicates that our dynamic model accurately captures the true contact behavior. Second, the RL-assisted policy significantly reduces force and torque tracking errors compared to the PD-only baseline.

V. Conclusion

This letter introduced a novel hybrid approach for robotic manipulation tasks involving precise object pushing. The key contributions include: (1) a comprehensive contact dynamic model featuring unilateral constraints and a box friction model, accurately capturing nonlinear, multi-contact frictional dynamics; (2) an ICR-based trajectory formulation that enables smooth and physically feasible motion between poses; and (3) a residual velocity policy that augments the baseline controller using a force-level training objective in simulation, while requiring no external force sensors at deployment. In both simulation and hardware experiments, the RL assistance approach achieved lower object translational and rotational tracking errors compared to a PD-only controller. Force/torque analysis further indicates a small gap between simulation and real measurements, supporting the accuracy of the dynamic model. Future extensions may explore generalization to a wider range of contact configurations (e.g., more contact points) and assess modeling limitations for tall or top-heavy objects subject to out-of-plane effects.

References

[1] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. A high-fidelity experimental dataset of planar pushing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 30–37.

[2] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *Int. J. Robot. Res.*, vol. 15, no. 6, pp. 533–556, 1996.

[3] K. Miyazawa, Y. Maeda, and T. Arai, "Planning of graspless manipulation based on rapidly-exploring random trees," in *Proc. 6th IEEE Int. Symp. Assem. Task Planning, From Nano Macro Assem. Manuf.*, 2005, pp. 7–12.

[4] S. M. LaValle and J. J. Kuffner Jr., "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, 2001, doi: 10.1177/02783640122067453.

[5] G. Lee, T. Lozano-Pérez, and L. P. Kaelbling, "Hierarchical planning for multi-contact non-prehensile manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 264–271.

[6] T. Yoshikawa and M. Kurisu, "Indentification of the center of friction from pushing an object by a mobile robot," in *Proc. IEEE/RSJ Int. Workshop Intell. Robots Syst.*, 1991, pp. 449–454.

[7] J. Zhou, R. Paolini, A. M. Johnson, J. A. Bagnell, and M. T. Mason, "Aprobabilistic planning framework for planar grasping under uncertainty," *IEEE Robot. Automat. Letters*, vol. 2, no. 4, pp. 2111–2118, 2017, doi: 10.1109/LRA.2017.2720845.

[8] M. Bauza and A. Rodriguez, "A probabilistic data-driven model for planar pushing," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 3008–3015.

[9] M. Bauza, F. R. Hogan, and A. Rodriguez, "A data-efficient approach to precise and controlled pushing," in *Proc. Conf. Robot Learn.*, 2018, pp. 336–345.

[10] M. Yang et al., "Sim-to-real model-based and model-free deep reinforcement learning for tactile pushing," *IEEE Robot. Automat. Lett.*, vol. 8, no. 9, pp. 5480–5487, Sep. 2023.

[11] C. Chi et al., "Diffusion policy: Visuomotor policy learning via action diffusion," *Int. J. Robot. Res.*, vol. 44, pp. 1684–1704, 2025.

[12] L. Cong, M. Grner, P. Ruppel, H. Liang, N. Hendrich, and J. Zhang, "Self-adapting recurrent models for object pushing from learning in simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5304–5310.

[13] N. Dengler, D. Großklaus, and M. Bennewitz, "Learning goal-oriented non-prehensile pushing in cluttered scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 1116–1122.

[14] S. Jin, R. Wang, M. Zahid, and F. T. Pokorny, "How physics and background attributes impact video transformers in robotic manipulation: A case study on planar pushing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 7391–7398.

[15] P. Lötstedt, "Mechanical systems of rigid bodies subject to unilateral constraints," *SIAM J. Appl. Math.*, vol. 42, no. 2, pp. 281–296, 1982.

[16] F. Gholami, M. Nasri, J. Kövecses, and M. Teichmann, "A linear complementarity formulation for contact problems with regularized friction," *Mechanism Mach. Theory*, vol. 105, pp. 568–582, 2016.

[17] F. Gholami, M. Nasri, J. Kovecses, and M. Teichmann, "A fast algorithm for contact dynamics of multibody systems using the box friction model," *J. Comput. Nonlinear Dyn.*, vol. 12, no. 1, 2017, Art. no. 011016.

[18] H. Olsson, K. J. Åström, C. C. De Wit, M. Gäfvert, and P. Lischinsky, "Friction models and friction compensation," *Eur. J. Control*, vol. 4, no. 3, pp. 176–195, 1998.

[19] J. Moorehead, S. Montgomery, and D. Harvey, "Instant center of rotation estimation using the Reuleaux technique and a lateral extrapolation technique," *J. Biomech.*, vol. 36, no. 9, pp. 1301–1307, 2003.

[20] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Mach. Syst.*, vol. MMS-10, no. 2, pp. 47–53, Jun. 1969.

[21] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[22] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *J. Dyn. Syst., Meas., Control*, vol. 103, no. 2, pp. 126–133, 1981.

[23] N. Hogan, "Impedance control: An approach to manipulation," in *Proc. Amer. Control Conf.*, 1984, pp. 304–313.

[24] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, "Residual policy learning," 2018, *arXiv:1812.06298*.

[25] F. J. B. Cıścar and F. M.-H. Maciá, "Trajectory planning using oscillatory chirp functions applied to bipedal locomotion," in *Proc. Int. Conf. Inform. Control, Automat. Robot.*, 2007, vol. 3, pp. 70–75.

[26] S. Chen and J. T. Wen, "Industrial robot trajectory tracking control using multi-layer neural networks trained by iterative learning control," *Robotics*, vol. 10, no. 1, 2021, Art. no. 50.

[27] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 23–30.

[28] O. M. Andrychowicz et al., "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 3–20, 2020.

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

[30] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 3207–3214.

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

[32] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marıń-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognit.*, vol. 47, no. 6, pp. 2280–2292, 2014.